Automation in Construction 171 (2025) 106004

Contents lists available at ScienceDirect

Automation in Construction

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

Design Healing framework for automated code compliance

Jiabin Wu *, Stavros Nousias, André Borrmann

Chair of Computational Modeling and Simulation, School of Engineering and Design, Technical University of Munich, 80333, Munich, Germany

ARTICLE INFO

Keywords:
Building information modeling (BIM)
Automated compliance checking (ACC)
Design space exploration (DSE)
Sensitivity analysis (SA)
Graph theory

ABSTRACT

Automated Compliance Checking (ACC) techniques have advanced significantly, enabling designers to evaluate building designs against codes. However, architectural engineers have to improve the design by manually implementing the ACC results, which is laborious, iterative, and requires domain expertise. To address this challenge, this paper introduces a Design Healing framework that adapts the original design into code-compliant alternatives. By integrating design data and ACC results, the framework identifies critical non-compliant components through a graph-based topological algorithm and sensitivity analysis. A prior knowledge-informed design space exploration is conducted to find valid alternatives and quantify modifications using weighted Euclidean distances, allowing designers to select options closely aligned with the initial design. Multi-scenario experiments demonstrate the framework's effectiveness in resolving architectural spatial design violations. By automating post-checking design adaptation, the framework reduces manual revisions and provides an efficient tool for achieving compliance while accommodating varying design constraints. This paper establishes a basis for advancing designer-centered automated design correction methods based on ACC techniques.

1. Introduction

There is a growing need to transition to more efficient workflows in the Architecture, Engineering, and Construction (AEC) industry. Of particular importance for the safety and performance of the later building is the (early) design phase, where fundamental decisions are taken [1,2]. To ensure the design quality and the safety of the later occupants, building designs are evaluated against various standards laid down in building codes, construction practices, and requirements from owners. The Building Information Modeling (BIM) development exploits semantically rich representations of buildings under design or construction [3,4], and offers a computer-interpretable form of building designs. Using BIM to automate design checking against regulatory obligations has been a research focus for many years [5]. Despite ongoing challenges, the checking methods and tools available today allow architectural engineers to evaluate the BIM models with many types of regulations in an almost completely automated manner [6].

Automated Compliance Checking (ACC) handles heterogeneous design models and design requirements to evaluate the compliance of building designs against building codes [7,8]. The generated checking results are typically presented in a textual format, at best represented by instances of the BIM Collaboration Format (BCF) that integrates textual descriptions with screenshots of the identified issues. Code checkers, such as Solibri Office [9], can offer explanations for checking failures and propose solutions. However, this capability is limited to simple changes, such as adjustments to property values, and does not extend to addressing more complex design challenges. Additionally, the code checkers provide suggestions in textual form, such as recommending enhancements to the fire-rating level of specific building elements. While these proposals are informative, they require further intervention by designers to implement. Consequently, building designers, typically architectural engineers, must interpret the ACC results and manually search for compliant solutions [10]. Correcting one issue could lead to violating another rule, potentially resulting in endless iteration cycles. This becomes a significant challenge considering the large number of building regulations that have to be taken into account.

Recent design support studies have explored the development of design alternatives to meet requirements in several areas, such as fire protection [11], interior room design [12], and quantity take-off (QTO) [13]. However, these approaches mainly focus on modeling domain knowledge as generation rules. The direct application of ACC results to support code-compliant design searches has not been investigated yet. Furthermore, these studies have primarily investigated code compliance on specific elements or spaces, with less attention to resolving complex design issues involving a multitude of elements and spaces.

* Corresponding author.
E-mail address: j.wu@tum.de (J. Wu).

https://doi.org/10.1016/j.autcon.2025.106004
Received 17 September 2024; Received in revised form 14 January 2025; Accepted 20 January 2025
Available online 28 January 2025

To overcome the aforementioned challenges, this paper proposes an approach that supports architectural engineers by performing automated searches for code-compliant solutions while adhering to constraints derived from design experience, client requirements, and other boundary conditions not tied to code compliance. This approach allows users to save time and focus on the creative aspects of design tasks. Specifically, the paper introduces a framework called Design Healing to automatically reach code-compliant alternatives closely aligned with the original design topology based on ACC results. The framework is generic and supports design issue correction by integrating constraints defined by human designers. To reduce the complexity of this paper, the considered rules focus on architectural spatial violations, addressing requirements like minimum corridor width, room size, and area, defined by several clauses of the International Building Code (IBC) [14].

The relevant design variables are initially identified through a propagation algorithm that leverages graph structures, accounting for dependencies among building components and allocated constraints by the authoring side. The concept is to first stay local, i.e., modifications without altering the topological connectivity and close to the source of the code violation, but if feasible solutions cannot be identified, the scope is progressively expanded to include more building elements for potential adjustments. A screen-based sensitivity method is applied to identify the most critical design variables, reducing the design space to a lower-dimensional subspace. Informed by prior knowledge, a conditioned Latin Hypercube strategy is utilized for directional and regional sampling of feasible areas. Finally, the weighted Euclidean Distance is calculated between newly generated variants to the origin to measure the degree of deviation, assisting in identifying suitable design solutions that are close to the initial design. This approach enables designers to incorporate code-compliant alternatives for further design refinements.

As a prerequisite, we assume the initial design is a parametric model, providing the necessary parametrics and realistic constraints to represent the initial design logic while allowing for adaptability. This paper builds on the tailored parametric capabilities of BIM authoring tools such as Autodesk Revit, Graphisoft ArchiCAD, Nemetschek Allplan, Vectorworks, and many others. It distinguishes itself from purely generative methods that rely on a fixed set of parameters and predefined design generation logic. The proposed Design Healing approach leverages fundamental design parameters – such as component dimensions and placement in the conceptual phase – as a robust foundation, allowing human designers to integrate their contextual knowledge through custom constraints into model adaptation.

We treat code checking as a "black box", using ACC outcomes as input for post-check design adaptations with a focus on building rules that can be translated and checked. The process of rule interpretation and execution is not the focus of this paper. The main objectives are as follows:

1. Develop a method that leverages ACC results to achieve code-compliant design alternatives closely aligned with the original design.
2. Establish a streamlined search approach to efficiently identify feasible solutions in the context of high-dimensional design modifications.
3. Formulate a robust representation of design variables and ACC results that captures dependencies and constraints, establishing a foundation for diverse real-world design correction scenarios.

The remainder of the paper is organized as follows: Section 2 introduces the background and related work based on the identified gap and challenges. The proposed method is described from a theoretical point of view in Section 3. Section 4 focuses on the experimental setup and proof-of-concept results. Section 5 discusses the contribution and limitations, while a conclusion is presented in Section 6 with directions for future research.

2. Background and review of related studies

Extensive research has been conducted in ACC to automate the regulatory requirement checking process. The ACC processes can be structured into four main components: rule interpretation, building model preparation, rule execution, and checking result reporting [7]. Different code compliance checking systems are developed to interpret the rules for checking purposes [15], forming the basis for automated processes. Diverse methods, such as Natural Language Processing (NLP) [16,17], prompt engineering [18], and other data-driven methods [19], have been employed to transfer regulatory information into computer-readable rules and enable automated code checking. Pauwels et al. [20] compare three approaches for semantic rule-checking and provide a performance benchmark to aid in adopting such approaches. The development of code checkers has shifted the paradigm from predominantly manual checks to automated design checking [21,22]. Soliman-Junior et al. [23] survey the designers' views on using automation to support regulatory compliance, highlighting the need for automation to support decision-making and minimize design rework. Amor and Dimyadi [6] review the evolving ACC approaches, noting that the community has arrived at a point where realized checking systems provide checks against building codes and standards. Despite these advancements, further research is needed in building object libraries and BIM tools for correct model generation.

Based on ACC, several studies investigate approaches for generating code-compliant designs. Lee et al. [11,24] have pioneered a design support system that provides design alternative recommendations to align with regulatory standards. Nevertheless, this system emphasizes semantic issues of fire-related objects, overlooking geometric and topological design challenges that might need iterative rectification. Liu et al. [13] have proposed a framework for auditing BIM models to achieve code-compliant quantities. Despite its contribution, this study focuses solely on semantic data integrity for QTO purposes. Similarly, Sydora and Stroulia [12] developed algorithms to automatically arrange elements in interior design spaces, producing valid alternatives that adhere to predefined design generation rules, though limited to placing objects within initially empty models. This scope excludes spatial relationships, which are typically incorporated to enhance rule-checking efficiency [25]. For instance, topological spatial relationships are extracted from the design model to support ACC [26]. Moreover, those existing methods are tightly constrained by specific requirements, with limited development of a theoretical framework for utilizing ACC results to inform extensive design modifications. Each design change – whether creating, deleting, or modifying entities – can trigger unintended violations in other parts of the model. To address this, Eastman et al. [27] propose a "patching" method to correct side effects and preserve design integrity. However, while this approach focuses on dependency relations within the design, it does not investigate the potential of leveraging ACC results to enhance design adaptability.

In summary, existing ACC-related design support studies focus on individual elements or spaces, which is inadequate for correcting complex spatial design issues. Correcting design issues across interrelated elements and spaces remains challenging. A streamlined post-checking workflow to connect ACC outcomes directly to resolving compliance violations remains lacking. Addressing this gap requires overcoming the following primary challenges:

1. Formulating a robust, computationally accessible representation to capture non-compliance-related components within the BIM model.
2. Investigating pragmatic parametric foundations to manage interdependencies and enable flexible adaptability in real-world design scenarios.
3. Creating streamlined design search approaches to achieve code-compliant alternatives that meet interrelated requirements.

Aligned with these challenges, this section reviews underpinning studies on (1) design representation techniques, focusing on graph-based representation of building design data (Section 2.1), (2) BIM-based parametric modeling that supports flexible design modification (Section 2.2), and (3) approaches and strategies for design alternative search (Section 2.3).

2.1. Graph representation of building design data

The BIM-based building design continuously integrates and updates the design knowledge into the digital model. Graph structures are widely employed for knowledge representation to manage complex design information and their interrelationships [28]. Isaac et al. [29] use the graph theory to represent, manage, and analyze building designs, defining dependencies between groups of building components to develop more efficient tools, such as design modularization. Khalili and Chua [30] present a graph data model to use semantic information for extracting and analyzing topological relationships among 3D building elements. Donato [31] integrates a space connectivity graph to evaluate design solutions based on connectivity. Graph approaches transform a building design into a network of nodes and edges, enabling effective representation and analyses of semantic and topological design data.

The graph representation can leverage the semantic data for design models in ACC processes. Conceptual graphs are used to unambiguously describe the regulatory knowledge [32] and to transfer it among actors in code compliance checking [33]. Xu and Cai [34] presents a semantic approach for automated compliance checking of underground utilities, utilizing graph-based knowledge representation to integrate heterogeneous geospatial and textual data. Additionally, Peng and Liu [35] transform specification provisions into a human-machine-readable knowledge graph, which is essential for reviewing BIM models. However, graph representation is primarily used to analyze design requirements from building regulations, with less focus on identifying building components that may be associated with design issues or facilitating design modifications. To support decision-making in design phases, Mattern and König [2] use graph data models to describe and manage the resulting models. The graph method efficiently derives design alternatives from the initial model by providing information about affected elements and interdependencies. Therefore, graph representation can be used to analyze the potentially affected components of ACC results, especially for design issues impacted by multiple elements and spaces.

2.2. BIM-based parametric modeling

BIM-based parametric modeling is pivotal in achieving well-aligned design outcomes that fulfill the intended design objectives [36]. Sacks et al. [37] outline essential considerations for building design automation through parametric modeling. Leveraging building design knowledge, architects employ parametric design systems to formulate architectural elements and rules [38], facilitating design improvements by adjusting parameters and their interrelationships. Furthermore, parametric design modeling can be classified into different types based on functional purposes. Generative design (GD) parametrics rely on mathematical rules with adjustable variables, providing flexibility and precision in design generation [39]. In early conceptual design phases, architectural design tools utilize sliders with defined specific data types, and value ranges to control design elements intuitively [40]. On the other hand, design parametrics in BIM systems can involve variables that control the properties, dimensions, and placement of design elements, ensuring precise object information and spatial relationships [4, 41].

GD systems, informed by parametric layouts and prescriptive knowledge from user feedback, exemplified in the pharmacy redesign case [42], can offer dynamic adaptability into design models. However, generative design approaches are usually investigated in downstream workflows [43] and rely on advanced parametric systems, which may not always be feasible in actual building projects. This is mainly because, for building engineers, it is more cumbersome to first create a full system of parametric model elements, constraints, and dependencies covering a space of potential solutions than directly working toward a building design. Yet, establishing an effective representational framework for such systems remains a challenging task [44]. In consequence, despite the expressive power of GD, it is applied to a limited extent in today's building design workflows, rendering its application less beneficial for correcting building designs undergoing code compliance checks.

Diverging from fully closed parametric designs within the GD paradigm, BIM models with necessary parametrics and constraints representing the design logic – as commonly found in today's practice – should act as the starting point for design alternative searches to achieve code compliance. The design models created by the designers may need to align with the extensive possibilities offered by generative methodologies. Therefore, it is crucial to develop a nuanced representation system to facilitate the formalization and effective exploration of design alternatives. In this paper, parametrics refer to those implemented by BIM authoring tools like Autodesk Revit, ArchiCAD, and Nemetschek Allplan rather than those used in mechanical CAD (CATIA, NX, Inventor) or geometric modelers like Rhino3D.

2.3. Design alternative search and exploration strategies

Leveraging the design space for alternative search is an effective method for addressing uncertainties in building design. Initiated on any parametric modeling problem, a design space encompasses all possible alternatives within a high-dimensional hyperbox, whose dimensions and corresponding intervals represent the value sets of choice variables [45]. Within a specific design topic, a design space is a multidimensional realm of design variable combinations [46]. Design space methodologies empower designers to directly interact with design variables, including different variable types such as booleans, numeric, and category, to achieve the design objectives [47].

Parametric modeling enables architects and engineers to embed their domain expertise and conduct design space exploration (DSE) [48, 49]. Based on multi-disciplinary design optimization, Gerber et al. [50] improve the design space generation and evaluation for energy design alternatives. Performance-based parametric modeling in architecture assists in determining variable importance during model setup [51]. Singh et al. [52] develop a web tool that supports early-stage design decisions by providing relevant information and surveys to get quantitative feedback from the users on DSE. Nevertheless, DSE approaches present significant challenges, such as the need for a large number of samples to tackle high-dimensional problems with continuous numerical input variables [45].

Given the complexity of building design, which typically involves numerous design variables, finding suitable alternatives could lead to computational overheads. The range of sampled variables is a crucial factor in defining the scope of the design space [53]. Additionally, a formal space representation and methods for navigating distinctive solutions must be developed to enable efficient exploration [54]. Instead of relying on a dense, comprehensive sample encompassing the entire space, iterative refinement of the DSE through directional adjustments and targeted zooming into promising regions is advocated [46]. The aforementioned exploration strategies should be investigated to enable an efficient design alternative search process.

To facilitate DSE, it is essential to identify key parametric variables. Sensitivity analysis (SA) is a widely used technique to reduce the complexity of high-dimensional design spaces by describing the input variables' contribution to the design's total performance [55]. SA methods are often categorized as local sensitivity analysis (LSA) or global sensitivity analysis (GSA) [56]. LSA uses the one-at-a-time

Table 1
Notation of essential symbols.

Symbol	*Description*
$\mathcal{R}$	The code checking result of one building rule
S	A design space
D	A design solution
C	A set of design constraints
X	A set of design variables
$\hat{X}$	A set of sensitive design variables
$G(V, E)$	G: graph, V: node, E: edge
$\mathbf{H}_j$	The weighted Euclidean Distance from the jth design variant D_j to the origin
y	The computation criterion of compliance checks
y'	The adapted computation criterion of compliance checks
$Y(y')$	The code checking result for sensitivity analysis

(OAT) approach to assess output variability by changing one input variable at a time [57], whereas the GSA method estimates the effect of an input variable on the output by varying all other variables. Some of the main outcomes from SA studies are [58]: (1) to obtain insights about the essential input variables (factor prioritization); (2) to receive information about the least influential variables that can be confidently fixed (factor fixing); (3) to know whether a directed input change provokes an increase or decrease in model output (sign of factor change).

The choice of the appropriate SA method depends on the technique's computational complexity. Comprehensive reviews have been conducted to compare the efficiency of various SA methods, guiding the selection of appropriate techniques [59,60]. The variance- and screening-based SAs are two promising methods in building analyses. With a variance-based GSA method, Singh and Geyer [61] analyze the parametric energy model and calculate the uncertainty contribution of each parameter at the early design stage. The variance-based SA methods, such as the Sobol method [62,63], can obtain stable results for non-monotonic and nonlinear problems [64]. The variance-based methods comprehensively assess the sensitivity by evaluating single and multiple variables' contribution to output variability [65], requiring a large number of model executions. It has been underscored that the screening-based method, particularly the Morris method, is an effective alternative for prioritizing parameter ranking to more resource-demanding SA techniques [66,67]. Moreover, the Morris screening-based method can identify whether input changes increase or decrease output, yet its potential to accelerate design space exploration processes remains underutilized.

3. Design healing framework

This section details the proposed Design Healing framework, which revises initial designs that violate building regulations into code-compliant alternatives while preserving alignment with the original design topology. The framework assumes as a prerequisite that the initial model is parametric, with essential parameters and constraints defined by the designers. The Design Healing framework comprises three main components (Fig. 1). Firstly, ACC results are analyzed to locate design issues, using a graph-based topological propagation approach to identify related building elements and design variables. Secondly, a Morris screening-based SA identifies key design variables linked to compliance violations, reducing the original design space. Finally, conditioned Latin hypercube sampling (cLHS) is applied in a multi-step DSE to guide the search toward compliant regions, with weighted Euclidean Distance quantifying deviations from the initial design to identify valid, similar alternatives. The source code is available online.[1] A notation table summarizes the essential symbols used in this paper (Table 1).

[1] https://github.com/Jaaaaabin/DesignHealingWu

3.1. Representation of building designs and compliance checks

3.1.1. Representation of building design data

To establish the groundwork for a computationally efficient adaptation approach, we propose using a structured representation of building designs. A design solution is symbolized as D, wherein every specific design is characterized by a collection of variables $X(x_1, \ldots, x_{i_{max}})$ and a set of associated design constraints $C(c_1, \ldots, c_{r_{max}})$. A design solution is described as $D(X, C)$, and the design space S consists of all possible design solutions. X enables explicit specification of the design's geometric, locational, and semantic features. For instance, a locational design variable denotes the geometric distance between an interior wall and the reference grid within the BIM authoring tool, as illustrated in Fig. 2. Besides, building elements engage in intricate interactions within the design model, typically supported by design constraints C to maintain their relational integrity. Thus, an initial design without specific constraints $C = \emptyset$ can be represented as $D((x_1, \ldots, x_{i_{max}}), C)$. By including only one constraint $c_1 : x_1 = 0$, the design can be written as $D((x_1, x_2, \ldots, x_{i_{max}}), c_1)$ with one constrained variable x_1. Design constraints can incorporate single or multiple variables. For illustrative purposes, consider a case where the objective is to maintain the minimum area for a specific room in the building. In this context, c_1 might indicate the alignment of an exterior wall with the building's outline grid.

Tailored user demands, regulatory requirements from building codes and regulations, and construction methods impose distinct limitations on design. In this paper, the design constraints C are specifically utilized to represent all the restrictions that do not originate from building codes. These constraints arise from a design-focused perspective, capturing elements that may limit potential modifications within the BIM model. Formalized as equations or inequalities, these constraints are conveyed through design variables. For example, constraints on location variables mean a wall with a fixed position has a specified location variable.

3.1.2. Representation of ACC results

Compliance checks evaluate whether a design adheres to the building regulation, resulting in informative checking result $\mathcal{R}$. For a given initial design D_{ini}, the obtained result is denoted as $\mathcal{R}(D_{ini})$. The checking feedback $\mathcal{R}$ can be categorized into two levels: *compliance level* $\mathcal{R}_c$, and *assessment level* $\mathcal{R}_a$. The compliant-or-not feedback $\mathcal{R}_c$ indicates whether the design meets specific regulatory requirements. For example, if the minimum area required for a room is 5 square meters and the evaluated room is only 4.3 square meters, then $\mathcal{R}_c$ would label the design as "fail" or "non-compliant", providing a binary evaluation. Additionally, code checkers usually report a brief assessment feedback $\mathcal{R}_a$ describing the required v_{req} and actual values v_{act} of the relevant checking requirement for both quantitative and qualitative rules. Expanding on the room area scenario, $\mathcal{R}_a$ specifies, "The space is 4.3 square meters, while the minimum area requirement is 5 square meters".

It is important to note that extracting and reasoning about textual information in ACC results, which involves NLP techniques, is out of

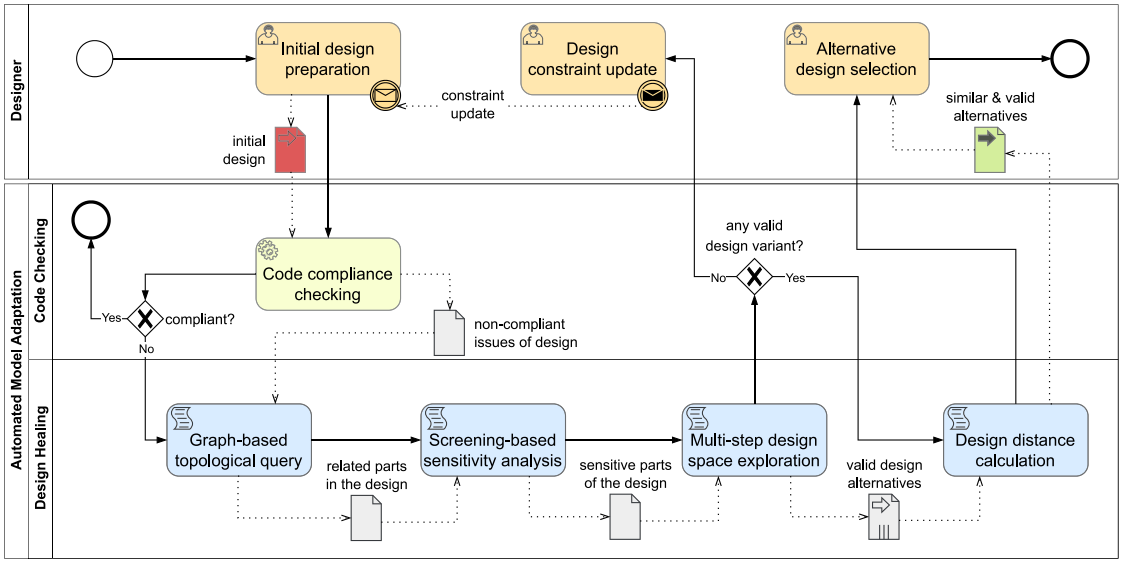


Fig. 1. Design Healing framework for automated code-compliant model adaptation: Step (1) graph-based topological propagation for finding relevant building objects to overcome non-compliance, Step (2) screening-based sensitivity analysis of design variables, and Step (3) multi-step design space exploration to achieve code-compliant alternatives.

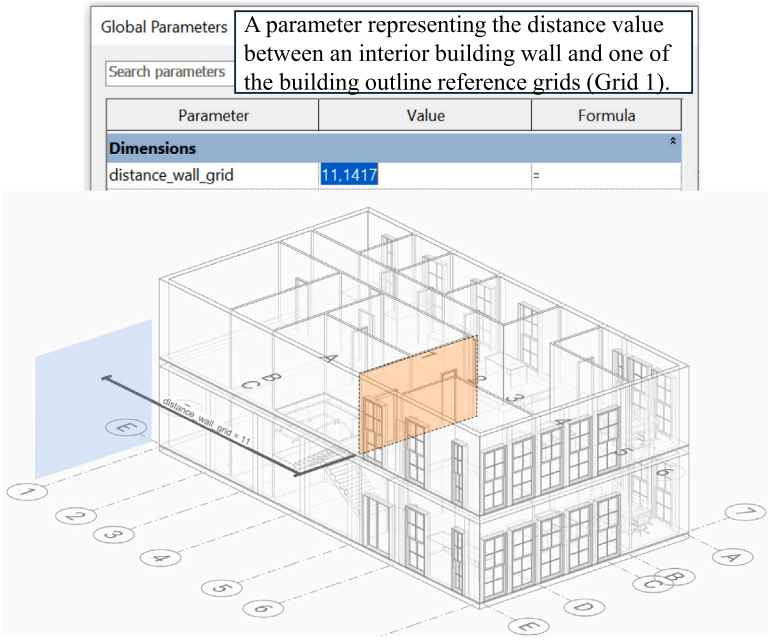


Fig. 2. Design variable for the distance between a wall and the outline reference grid.

the scope of this paper. We implement a mimic code checker using the BIM authoring tools' Application Programming Interface (API) to directly return both the required v_{req} and actual values v_{act} of relevant requirement. To ease the use of the code-checking results for design modifications, a compliance computation criterion y is established, as outlined in Eq. (1).

$$y = \lambda \left| \frac{v_{act} - v_{req}}{v_{req}} \right|, \text{ where } \lambda = \begin{cases} 1, & \text{if } \mathcal{R}_c(D) = \text{true} \\ -1, & \text{if } \mathcal{R}_c(D) = \text{false} \end{cases} \tag{1}$$

This criterion applies to situations where a specific regulation is evaluated a single time for a design solution D. The directional factor λ aligns the sign of y with the compliance conditions. Building on the mentioned minimum room area scenario, suppose the required minimum area for a room is 5 square meters, but the actual area is 4.3 square meters. This situation would result in a negative y, indicating non-compliance. Conversely, if the actual room area exceeds 5 square meters, it yields a positive y, denoting code compliance.

Directly analyzing the compliance criterion y can be challenging, as evaluating the same requirement across multiple model subparts may reduce its effectiveness due to consistently compliant sections. For instance, the minimum room area requirement may be met in most rooms except one, making the compliance calculation less attentive to the non-compliant part. To focus on non-compliant parts, a reduction coefficient β ($0 \leq \beta \leq 1$) is introduced (Eq. (2)). Intermediate values of β allow for partial inclusion of compliant elements. For instance, in a building with many rooms subject to minimum area checks, setting $\beta = 0$ excludes compliant rooms from the analysis, focusing only on those not meeting the minimum area requirement.

$$y' = \beta' y, \text{ where } \beta' = \begin{cases} \beta, & \text{if } \mathcal{R}_c(D) = \mathcal{R}_c(D_{ini}) = \text{true} \\ 1, & \text{else} \end{cases} \tag{2}$$

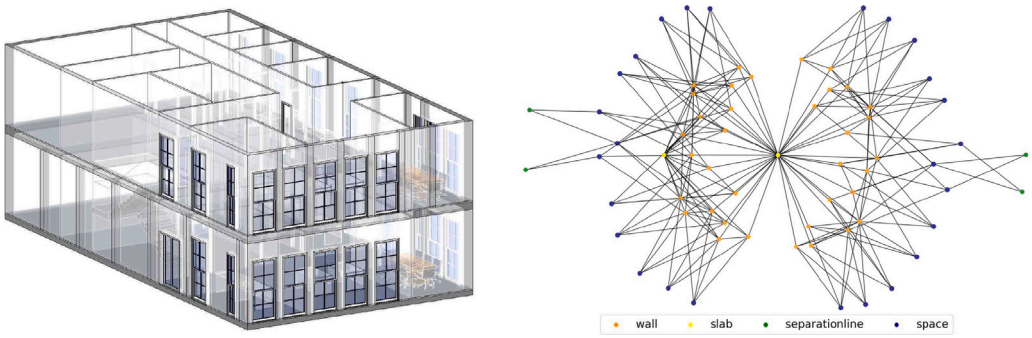


Fig. 3. Example of building design and corresponding topological connectivity graph.

3.2. Graph-based propagation of non-compliance

3.2.1. Graph representation

Compliance checks identify the design issues and report the related building objects where the non-compliance lies. However, other building parts may interact or share common attributes with the reported objects, yet they still need to be recognized. Thus, the graph techniques are leveraged to extract all potentially related building elements and spaces for the observed non-compliance. However, the choice of graph representation, including nodes and edges, depends on the nature of the rule requirements being investigated. For instance, when compliance checks multi-level conditions, the building graph might adopt a hierarchical structure to provide a suitable analysis structure. Similarly, adjacency and connectivity data are needed when checks relate to the building's spatial arrangement.

This paper focuses on adapting the design to spatial building regulations via localized design changes that preserve the original topological connectivity. To clarify the concept of "localized changes", consider the scenario where the width of a building's floor does not conform to the building regulations. Addressing this issue without altering the topological connectivity requires adjusting the floor's width, which may, in turn, lead to modifications in all connected walls. Therefore, a topological connectivity graph is used to capture the interdependencies among building elements and spaces for analyzing design alterations. A topological connectivity graph is presented in Fig. 3.

The automatic generation of graph nodes and edges is accomplished via the BIM authoring tool's API, converting BIM-based design data into a topological connectivity graph, denoted as G. By extracting building design objects and their properties from the design model, we establish a node set V for different building objects (elements of different types and spaces) and an edge set E representing topological connectivity relationships. If certain building objects have constraints, such as fixed locations assigned by designers, the corresponding graph nodes are labeled as constrained nodes V_c. All other nodes, representing unconstrained objects, are labeled as unconstrained nodes V_u. In addition, non-compliant objects identified from the ACC results are marked within the generated graph as the starting points (V_{ini}) for topological propagation to identify other relevant building objects.

3.2.2. Topological propagation of non-compliance

A graph-based topological propagation algorithm is designed to find the building objects that could affect code compliance. We exclusively use undirected edges, E_u, to represent topological connectivity relationships, specifically between different types of building objects (e.g., a room and its boundary walls). The topological propagation is detailed in Algorithm 1 in Appendix A. A trigger function, $\mathcal{T}(G(V, E), C)$, is employed to distinguish between constrained and unconstrained propagation steps.

As illustrated in the topological connectivity graph of building walls and spaces in Fig. 4, the algorithm performs a neighbor search from V_{ini}, traversing both unconstrained nodes V_u and constrained nodes V_c via undirected edges E_u. When the propagation reaches constrained building objects, the specified design constraints C are incorporated into the subsequent model adaptation. For instance, when an exterior wall is designed to align with a fixed outline grid, it is treated as a constrained node. C are numerically maintained through dependencies among related design variables, which restrict a subset of propagation trends. Consequently, the constrained nodes V_c (such as fixed exterior walls) are excluded from the variation set.

The algorithm progressively identifies related building objects concerning the detected compliance issues while preserving C. From a methodological perspective, propagation could extend to cover the entire building. However, assessing the full building for each design issue is unnecessary. Thus, once all relevant C from the authoring side are incorporated, the algorithm is stopped, defining the maximum propagation level l_{max}. By regulating l_{max}, the algorithm confines non-compliance issues to a localized area centered on the initial points of compliance check failures. This process produces a set of existing independent design variables, $X(x_1, \ldots, x_i, \ldots, x_{i_{max}})$, corresponding to unconstrained nodes V_u that are reached through topological propagation.

3.3. Factor prioritization using morris sensitivity analysis

The relationship between the design variables and the code compliance criteria among the building design is typically a non-formalizable and integrable nonlinear function $Y = f(X)$. For instance, satisfying the minimum room area (IBC 1207.3) requirement for multiple connected spaces returns a high-dimensional non-linear adaptation process. The design space expands exponentially to infinite potential outcomes when dealing with many independent design variables. Consequently, comprehensively exploring all conceivable options can result in significant computational overheads.

Benefiting from its ability to tackle non-linearities and identify the sensitivity signs [68], the Morris screening-based approach can be used for determining the essential design variables impacting the code compliance of the design. The input data for the sensitivity analysis against a single rule in the building codes can be expressed as in Eq. (3), where k_{max} represents the evaluation times of a rule on multiple portions of a model, such as the minimum room area evaluation on multiple rooms within a building design. Furthermore, the approach sums all evaluation results when conducting sensitivity analyses on design variables across multiple regulations.

$$X(x_1, \ldots, x_i, \ldots, x_{i_{max}}), \text{and } Y(y') = \sum_{k=1}^{k_{max}} y'_k \tag{3}$$

For the sake of completeness, the essential elements of the Morris screening-based sensitivity analysis are provided in Appendix B.

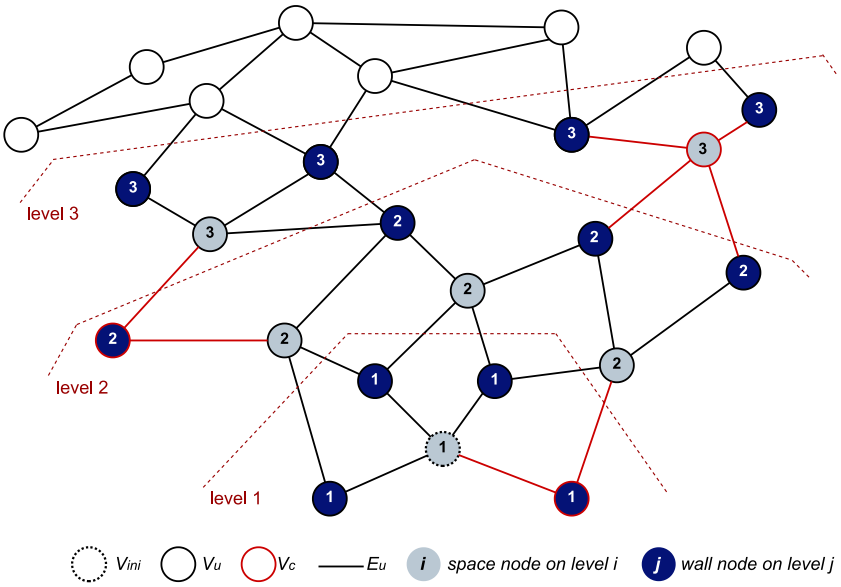


Fig. 4. Graph-based propagation of non-compliance at V_{ini} via topological connectivity relationships among building walls and spaces (exemplified with $l_{max} = 3$). The constrained graph nodes and their related edges are highlighted in red.

In the Morris method, the input variable space, X, consists of i_{max} number of input variables, x_i, each defined within a discrete range $\mathfrak{X}_i^{sa} = (x_i - \Delta x_i, x_i + \Delta x_i)$, normalized to a scale $(0, 1)$. μ_i and σ_i are two sensitivity measures that respectively represent the mean and the standard deviation of the distribution of the influence caused by x_i (Eq. (B.2)). μ_i assesses the sensitivity strength and influence trend of x_i on the compliance criteria Y considering all first- and higher-order effects that are associated with x_i, and σ_i indicates possible interactions with other variables: Compared to μ_i, the absolute-value-based version μ_i^* can provide an accurate importance measure [69], which is free of any non-monotonic input to output behavior present in μ_i.

Based on the sensitivity rankings of the variables presented by μ_i^*, the sensitive variables are sorted out from the design variables and taken in the subsequent space exploration steps. On the other hand, the signs of influence, illustrated by the sensitivity mean values μ_i, demonstrate in which direction the design variation contributes positively to improving the evaluation criterion (Eq. (2)). To elaborate, all the boundary walls of a room that is too small are logically relevant to the minimum area requirement. Adjusting the location parameter of these walls leads to wall displacement in different directions. Influence trends indicate whether each parameter change positively or negatively impacts requirement fulfillment, with sensitivity rankings and influence signs guiding the subsequent sampling process.

3.4. Design space exploration toward code compliance

Configuring the design space S encompasses all possible design variants, with complexity depending on each variable's range, covering aspects like architecture, spatial arrangements, and materials. The Design Healing approach seeks valid design alternatives that exhibit minor deviations from the original design, thereby requiring the design space S to focus only on alternatives with limited discrepancies from the initial design. As outlined in Section 3.1.1, the design is formally represented by variables X and associated constraints C. Accordingly, adaptation deviation can be calculated based on differences in both design variables and constraint levels. When considering design variants that obey identical constraints, we employ the weighted Euclidean Distance $\mathbf{H}_j$ to analyze the degree of deviation between the jth (newly) generated variant and the initial design $\mathcal{D}_{ini}$:

$$\mathbf{H}_j = \sqrt{\sum_{i=1}^{i_{max}} w_i (x_{ji} - x_{ini,i})^2},\ (i = 1, 2, \ldots, i_{max}; j = 1, 2, \ldots, j_{max}) \tag{4}$$

Among all variables involved in the design space exploration, a weight of $w_i = 1$ is assigned to the variable whose related building object has the most interconnections, with the weights of variables calculated proportionally. The $\mathbf{H}_j$ quantifies the difference between a new variant and the initial design by considering the varying influence of design variables. The $\mathbf{H}_j$ is intended to support designers in selecting valid alternatives that remain as close as possible to the initial design. From a design modification perspective, deviation depends on the magnitude of differences in design variables and the scope of design space dimensions affected by changes. Therefore, we analyze the factorial Euclidean Distances $\mathbf{H}_{j,i}$, which represent the absolute difference in each dimension $x_{j,i}$. This method provides insight into how $\mathbf{H}_j$ is distributed across relevant dimensions.

3.4.1. Space initialization

The initialization of the local design space relies on graph-based propagation of non-compliance. The topological propagation helps build the initial dimensions of the design space, as described in Section 3.2.2. To achieve this, the building objects reached by the propagation are collected, introducing the initial independent design variables X.

For design variables affected by the constraint set C, the constraints are incorporated via parameter dependencies to separate the dependent variables from the primary dimensions X. Consequently, the initial dimensions can be varied independently, while other constrained variables are adjusted automatically by maintaining the allocated constraints C. The initial design space is denoted as $S(X, C)$.

3.4.2. Space reduction

Design adaptation on all primary dimensions is anticipated to satisfy the regulatory requirements. However, exploring the entire initial space requires an extensive number of samples. As described in Section 3.3, the Morris SA helps to evaluate the effect of the variables on the model compliance by sampling in relatively small intervals in each dimension. The sampling interval $\mathfrak{X}_i^{sa}$ for each initial dimension is expressed as $x_{ini,i} + U(-\Delta x_i, \Delta x_i)$, where Δx_i is an arbitrary value defined experimentally to avoid violating the initial building topology, ensuring that the spatial relationships and key architectural design elements remain unaltered.

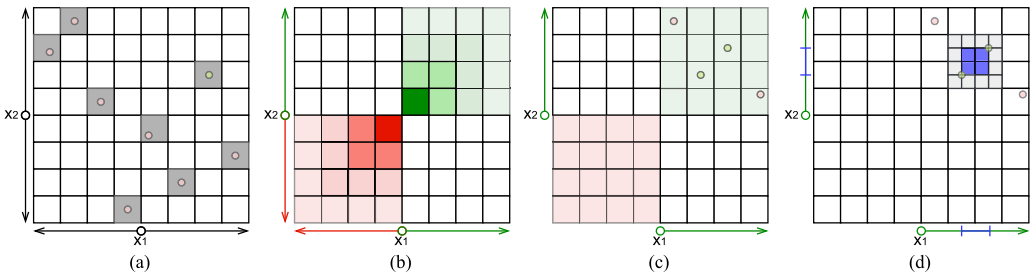


Fig. 5. Illustration of the conditioned Latin hypercube sampling with ancillary data: (a) Two-dimensional LHS centered at the initial values of variables x_1 and x_2; (b) Sensitivity signs of variables x_1 and x_2: the green area represents a positive influence, while the red area indicates a negative impact; (c) Conditioned sampling based on positively related ranges; (d) Conditioned sampling based on preliminary feasible ranges.

Based on the SA results (μ_i^*, μ_i, and σ_i), the unimportant variables are filtered out from the design space, reducing the space dimensions to $\hat{X}$ ($\hat{X} \subseteq X$). The space reduction process focuses on dimension reduction and prioritizes the sensitive aspects of the problem over others, helping to find feasible solutions for code compliance. Besides, the sensitivity signs indicate the influence trend of each variable, which is used as ancillary data for subsequent samplings.

3.4.3. Space enrichment

With the sensitivity rankings and signs, we employ a conditioned Latin hypercube sampling (cLHS) strategy to explore the design space progressively in potential code-compliant subspaces. The LHS strategy ensures a good scan of the inputs' variation by providing a uniform and comprehensive coverage of the design space [70]. It samples j_{max} variants from a set of i_{max} design variables, each following specific distributions. For each variable $x_i \in \hat{X}$, the sampling range $\mathfrak{X}_i$ is divided into j_{max} contiguous segments, aligned with the distribution associated with that variable. A value for the variable x_i is selected randomly from the interval $\mathfrak{X}_{ij}$ for $i = 1, 2, \ldots, i_{max}$ and $j = 1, 2, \ldots, j_{max}$. Then, the $i_{max} j_{max}$ values of all i_{max} variables are used to produce the ordered combinations, each of which is a sample with variables of $X_j = [x_{1j}, \ldots, x_{ij}, \ldots, x_{i_{max}j}]$, where $j = 1, 2, \ldots, j_{max}$. centered at the initial values of the variables x_1, x_2, a two-dimensional sampling is illustrated in Fig. 5. The distribution of samples across the design space maximizes the likelihood of capturing a diverse range of scenarios.

The cLHS method considers the existing ancillary data, incorporating prior knowledge from the SA results. Only the positively related part of $\mathfrak{X}_i^{sa}$ is selected initially for cLHS sampling on each variable x_i. Taking the sensitivity signs of each variable as the exploration trend, the sampling ranges $\mathfrak{X}_i$ are adapted progressively until a sufficient number of valid options are acquired. To ensure that the set of options is diverse enough without being overwhelming, selecting a sufficient number depends on multiple factors, including the sampling scope and the rule complexity. If the goal is to provide code-compliant alternatives for subsequent design steps, only a few valid variants are needed. Additional samples can help identify boundaries between valid and invalid spaces, though complex rules may limit valid options. Consequently, designers must control the degree of sufficiency. In cases where no feasible alternative can be reached, major intervention from designers is needed, which will change the design space configuration.

As shown in Fig. 5, the identified code-compliant alternatives constitute a preliminary feasible cluster, of which the derived contour is measured for potential valid ranges. We use the preliminary ranges to refine the design space, effectively navigating the exploration toward potentially valid sub-spaces (if any) within the formalized design space. Finally, the variants' deviation compared to the initial design model is measured in terms of $\mathbf{H}$ on all dimensions $\hat{X}$ of the final design space $\mathcal{S}$.

When design exploration fails to yield any code-compliant alternatives without breaching specified design constraints, expanding the design space by relaxing or adjusting certain constraints becomes essential. For instance, if two room boundary walls of a room are fixed, conflicts with neighboring rooms might prevent meeting minimum area requirements. Designers may need to release the constraints on non-structural elements in such cases. This results in a more significant deviation from the initial design.

4. Case study and results

This case study demonstrates the applicability of the proposed framework. To illustrate the proposed method, a multi-story building model was used as the initial design, with global overall dimensions of 38 by 19 m. The experiments are conducted on the ground floor (Fig. 6). In this model, we employ a basic set of spatial variables concerning the distances between wall elements and the building's grid lines—specifically, Grid A in the east–west orientation and Grid 1 in the north-south direction. The experiment focuses on the spatially related building rules outlined in Chapters 10 and 12 of the 2018 International Building Code [14]. The selected rules examine spatial building requirements, including corridor width and capacity requirements (IBC 1020.2), minimum room widths (IBC 1207.1), and room area (IBC 1207.3). Dynamo for Autodesk Revit was employed to implement code checker [71].

The initial design is positioned in the transitional phase between the Concept Design and Spatial Coordination stages, according to the Royal Institute of British Architects (RIBA) work plan [72]. In the Concept Design phase, the architectural concept is prepared by incorporating strategic engineering requirements and undertaking the design reviews from the client and project stakeholders. After agreeing on the route to code compliance in the Concept Design phase, the design is reviewed against the building regulations at the end of the Spatial Coordination stage. The architectural and engineering information is spatially coordinated in the Spatial Coordination stage.

To reflect real-world design scenarios where design models possess reduced parametric information, only the distances between building components and outlines are considered (see Fig. 2) as initial input variables. We assume that the parametric system is configured with the necessary parametrics and constraints to capture the boundary conditions and spatial dependencies intended by the designer. Unlike the fully developed parametric systems found in GD approaches, this basic set of spatial variables is more suited to design models used for code compliance checks. It is important to note that this paper does not cover parameters used in architectural conceptual design tools like Rhino/Grasshopper.

Following the RIBA Concept Design phase, where fundamental architectural concepts like the building outline and layout are established, these concepts are anticipated to serve as a basis for subsequent design processes. We use initial constraints C_{ini} to represent the conditions to retain. This case study delineates the model adaptation problem into two distinct scenarios: a highly constrained case and a case with updated constraints. In both scenarios, the positions of the atriums and stairways are firmly established, forming an integral part of the finalized design plan originating from the Concept Design

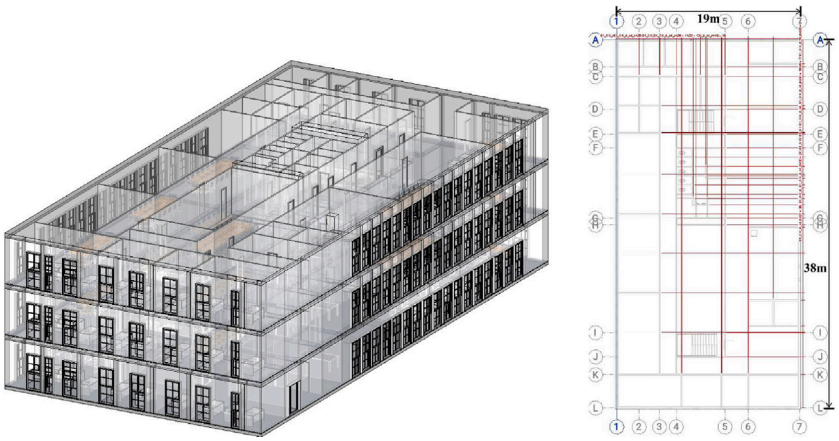


Fig. 6. Exterior appearance (left) and 2D plan of the ground level with dimensions as design variables (right).

Fig. 7. Propagation results showcasing with one single propagation (IBC 1020.2, IBC 1207.1, and IBC 1207.3). Spaces subject to propagation are highlighted in yellow, while affected objects are marked in red. The walls associated with ew_6, sn_{21} and sn_{10} are identified for potential local variation. Conversely, the wall of the service space, denoted by ew_7, is designated to remain unchanged from the initial design.

phase. Additionally, allowing for complete variability in all aspects of the model could substantially deviate from the initial design layout or lead to unrealistic building layouts. Following the common design practice, we set the corridors to straight spaces to maintain the initial layout configuration. Those initial conditions C_{ini} are retained in the experiments. The highly constrained scenario is characterized by specific design constraints that impose limitations on design variations. Conversely, the updated scenario allows for greater design adaptability.

4.1. Scenario 1: a highly constrained case

In this scenario, the incorporated constraints maintain the precise position and configuration of service spaces within the whole building C_h. Thus, following the representation principle in Section 3.1, the highly constrained design is represented by $D_h((x_1, \ldots, x_{i_{max,h}}), (C_{ini}, C_h))$. The developed code checker reports the non-compliance on minimum corridor width (IBC 1020.2), minimum room width (IBC 1207.1), and minimum room area (IBC 1207.3).

Building data is extracted using the BIM authoring tool's API to create a topological connectivity graph with the Networkx library. Fig. 7 illustrates non-compliant issues, including a narrow corridor and a small office, which are designated as the starting nodes V_{ini} in the graph. As described in Section 3.2.2, the graph-based topological propagation is initialized from V_{ini}. In this highly constrained case, propagation is manually halted when it reaches the service spaces because the linked wall element (ew_7) is fixed in place. Consequently, the unconstrained propagated walls (related variables ew_6, sn_{21}, sn_{10}) are taken as the initial adaptation set.

In adherence to the initial building design topology, comprehensive sampling is performed based on the initial design variables (ew_6, sn_{21}, sn_{10}). This sampling, however, yields no valid design alternatives, indicating that no feasible solution exists within the initial constraint sets (C_{ini}, C_h). The design space derived from $D_h((x_1, \ldots, x_{i_{max,h}}), C_{ini}, C_h)$ does not encompass any code-compliant design solutions. In such a scenario where no valid alternatives are achievable, designer interventions are required to update constraint conditions. In our experiments, we simulate human intervention by partially relaxing C_h to streamline the exploration of feasible solutions.

4.2. Scenario 2: a case with updated constraints

Given the lack of feasible solutions in Scenario 1, the restriction on service spaces is updated by preserving only the total area of the service spaces. This adjustment allows varying the service spaces while keeping its total area. Accordingly, the updated design is represented by $D_u((x_1, \ldots, x_{i_{max,u}}), C_{ini}, C_u)$.

4.2.1. Initialization of design space via topological propagation

As Section 3.2.2 outlines, the graph-based propagation approach finds topologically connected building objects from the previously identified building objects on each propagation level. It propagates through

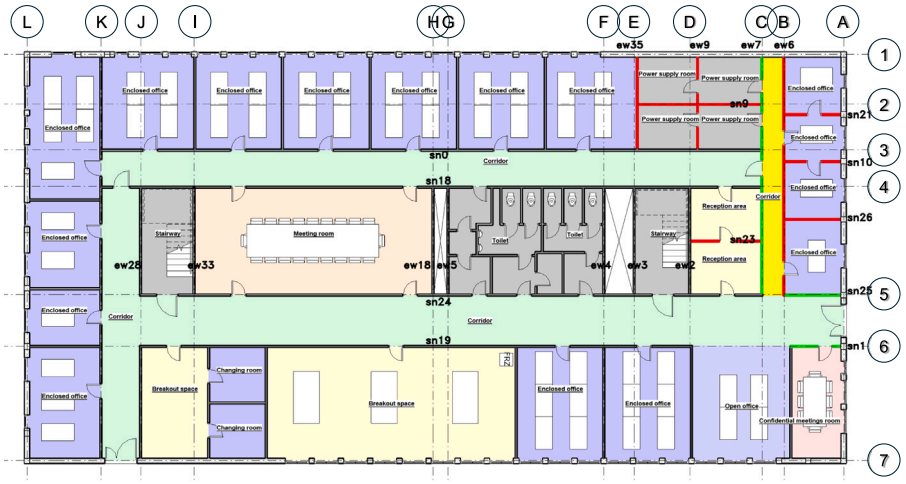


Fig. 8. Propagation results on the rule IBC 1020.2 with $l_{max} = 3$. Spaces subject to propagation are highlighted in yellow. Among the propagated walls, the constrained nodes V_c are marked in green, and the propagated unconstrained nodes V_u are marked in red.

the service spaces by keeping the associated constrained elements satisfying the specified set of constraints C_{ini} and C_u. The propagation continues until all building elements related to C_u are reached, yielding $l_{max} = 3$ for this scenario.

The design variables that are fixed by the initial constraints C_{ini} are labeled $(sn_0, sn_{18}, ew_{28}, ew_{33}, ew_{18}, ew_5, ew_4, ew_3, ew_2, sn_{24}, sn_{19})$ with their respective variable names in Fig. 8, with unconstrained nodes V_u and constrained nodes V_c are highlighted alongside the corresponding variable names. The movement of the corridor wall (related to variable ew_7) is synchronized with that of the boundary wall (related to variable ew_{35}) of the adjacent service spaces, maintaining the constraints C_u via parameter dependencies. Simultaneously, the variation of the propagated corridor walls (related to variables sn_1, sn_{25}) is constrained to maintain the standardized corridor shape. The variables $X(sn_{21}, sn_{10}, sn_{26}, ew_6, sn_9, sn_{23}, ew_9, ew_{35})$ are determined as the primary input based on the topological propagation, while the values of the dependent variables $X_{dep}(ew_7, sn_{25}, sn_1)$ correspondingly adhere to relationships defined via constraint sets C_{ini} and C_u. Doing so, the design variable candidates for SA are reasonably elected, enabling the reduction of the computational costs beforehand.

4.2.2. Reduction of design space via sensitivity analyses

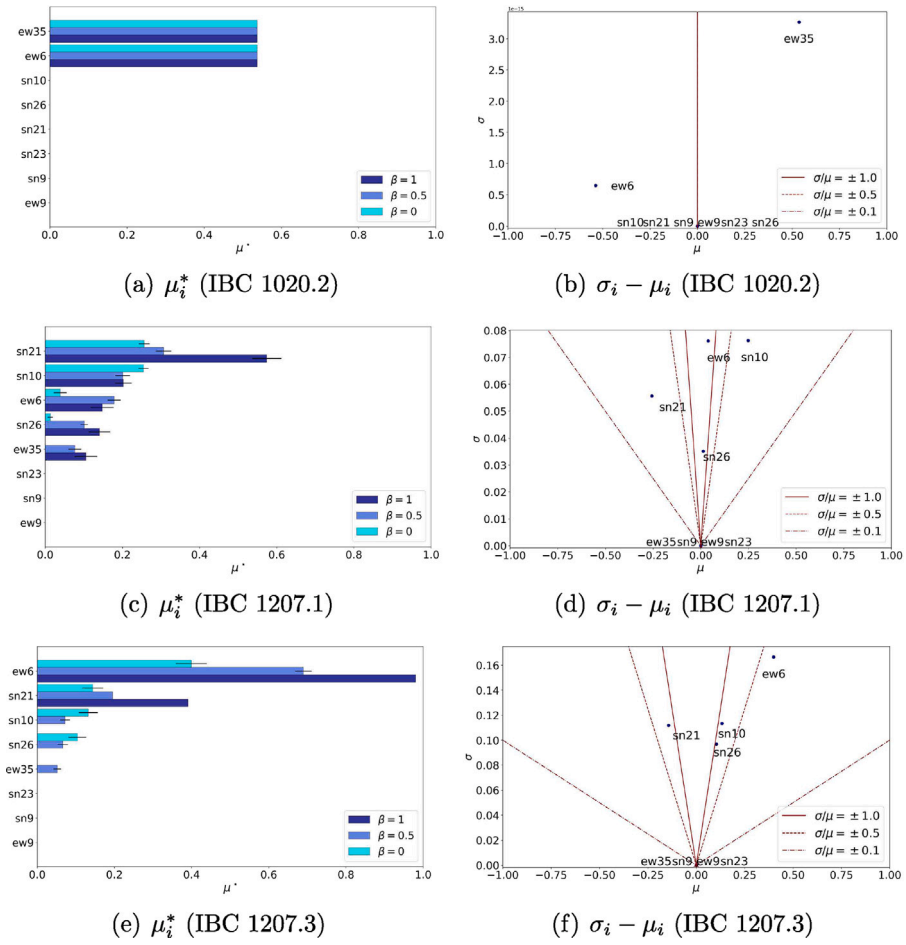
We conduct sensitivity analyses with $\beta \in \{0, 0.5, 1\}$. As detailed in Section 3.1.2, when $\beta = 1$, the effect of the constant compliance is interconnected. Conversely, this disturbance can be disregarded by setting $\beta = 0$. Thus, $\beta = 0$ is chosen to ignore the effects caused by the constantly compliant portion within the design. This approximation enables us to determine the essential components that drive transitions between non-compliance and compliance within the design. Based on the experimental results, the most effective SA parameters were selected. Each sampling spans six levels within the design variable space. A total of 200 Morris trajectories are generated, from which 64 trajectories with the greatest distance spread are selected for optimization.

To maintain the initial design topology, variation values are constrained to half the minimum distance between the issue-related grids — specifically, $0.9m$ between grids B and C. The SA sampling ranges are tested with $\Delta x_i^{sa} \in \{0.15\ m, 0.3\ m, 0.45m\}$. The three ranges yielded comparable results; therefore, only the results for $\Delta x_i^{sa} = 0.3m$ are presented. Building on the theoretical foundations outlined in Section 3.3, the Morris effects of the design variables are analyzed against each building rule (Eq. (B.1),(B.2)).

Design variables are ranked based on their absolute effect μ_i^*, with higher values indicating a stronger influence. The conducted SA shows that two variables, ew_{35} and ew_6, have a significant impact on compliance with IBC 1020.2 requirement (Fig. 9(a)). Similarly, the most influential input variables for compliance with IBC 1207.1 and IBC 1207.3 requirements are shown in Figs. 9(c) and 9(e). On the other hand, the non-linearity and interaction indices are collectively represented in the covariance visualization ($\sigma_i - \mu_i$ with $\beta = 0$), as illustrated in Figs. 9(b), 9(d), and 9(f). Points near the origin indicate non-essential variables, while those farther in the μ direction are generally important, and those farther in the σ direction represent variables with significant non-linearity in relation to other variables. For the IBC 1020.2 requirement, relatively low non-linearity is observed, as the minimum corridor width is influenced only by the positions of the two side walls. Strong interaction effects and significant non-linearity are detected for IBC 1207.1 and IBC 1207.3 requirements due to their recurring assessment of interconnected spaces. Therefore, solely relying on the sensitivity ranking is insufficient to effectively guide the subsequent sampling process.

To effectively support design space exploration, compliance with all relevant IBC rules is analyzed by summing the evaluation criteria. As illustrated in Fig. 10(a), the μ^* values present that the effects of design variables are detected with high confidence, suggesting strong reliability in the sensitivity rankings. The actual sensitivity effect, represented by the indices μ, provides a directional depiction of the impact of the related design variables (Fig. 10(b)). As outlined in Section 3.3, this actual sensitivity strength reveals whether a variable elicits a negative or positive effect when a change occurs from its initial value. However, it is noted that the determined directed influence of the variables, as indicated by the μ indices, introduces a degree of variability. This variability arises due to the inherent conflicts between selected IBC rules. For example, within the targeted region, meeting minimum corridor width requirements can conflict with minimum room area requirements, leading to shifts in variable influence as they work to balance compliance across multiple requirements.

Among the primary variables $X(sn_{21}, sn_{10}, sn_{26}, ew_6, sn_9, sn_{23}, ew_9, ew_{35})$, five influential variables, $\hat{X}(ew_{35}, sn_{21}, sn_{10}, ew_6, sn_{26})$, are identified as relevant for compliance with the three IBC rules under consideration. It reduces the design space to a 5-dimensional space, which is initialized via $\hat{X}$ by following the updated set of constraints C_u. Sampling with higher values of $ew_{35}, sn_{10}, sn_{26}$ positively affect the compliance results, while higher sn_{21} values causes significant negative effects. Although the absolute effect induced by ew_6 is significant according to μ_i^*, the sensitivity sign stays uncertain since a high degree of deviation σ_i is detected in regard to μ_i.

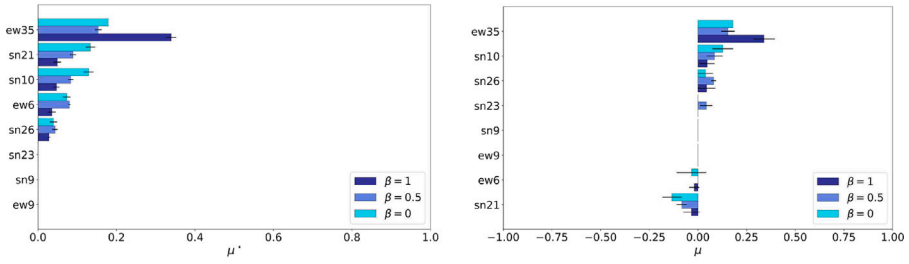


(a) μ_i^* (IBC 1020.2) (b) $\sigma_i - \mu_i$ (IBC 1020.2)

(c) μ_i^* (IBC 1207.1) (d) $\sigma_i - \mu_i$ (IBC 1207.1)

(e) μ_i^* (IBC 1207.3) (f) $\sigma_i - \mu_i$ (IBC 1207.3)

Fig. 9. Sensitivity results on individual rules: corridor width and capacity requirements (IBC 1020.2), minimum room widths (IBC 1207.1), and minimum room area (IBC 1207.3).



(a) Absolute influence μ_i^* (b) Directed influence μ_i

Fig. 10. Sensitivity rankings of the design variables for code compliance (three rules).

4.2.3. Multi-step design space exploration with prior knowledge

Building upon prior knowledge from Morris SA, specifically the signs of sensitivity indices, we use the cLHS to refine the design space toward code-compliant design alternatives. Following the description in Section 3.4.3, the cLHS method initially conforms to the positively influential segments of $\mathfrak{X}_{i,pos}^{sa}$. This approach, which is more informed than pure random strategies, leverages the results of SA to navigate efficiently toward valid sub-spaces if they exist. We incrementally expand the exploration ranges $\mathfrak{X}_i$ until feasible design solutions are attained. Subsequently, the exploration is adapted within the preliminary valid ranges derived from the previously identified feasible alternatives until a threshold number of valid variants is achieved. We take a threshold of 100 to provide sufficient code-compliant alternatives for the distance calculation between design variants. The sampling details and results of the multi-step cLHS are summarized in Table 2.

Among designs adhering to identical degree of constraints (C_{ini} and C_u), the degree of deviation from design variants D_j to the initial design D_{ini} is measured in terms of the weighted Euclidean Distance $\mathbf{H}_j$ and factorial Euclidean Distance $\mathbf{H}_{j,i}$, where i corresponds to the variables $x_{j,i} \in \hat{X}_j$. The weights of the variables $\hat{X}_j(ew_{35}, sn_{21}, sn_{10}, ew_6, sn_{26})$ are calculated based on the connectivity situation among building elements (excluding spaces) as $[0.6, 0.4, 0.4, 1.0, 0.4]$. The variable ew_6 has the most interconnections within the design space. Thus, a weight of $w_{ew_6} = 1$ is assigned, with weights for the other variables calculated proportionally $(0 < w_i \leq 1)$.

Based on the experimental results, code compliance among simulated design variants is achieved at $\mathbf{H}_j = 0.419$. Ordered by increasing weighted Euclidean Distance $\mathbf{H}_j$ for each sample, the simulation results are illustrated in Fig. 11. The upper part of the figure shows the accumulated number of valid design alternatives, N_{valid}, in relation

Table 2
The multi-step cLHS sampling details and results.

Sampling rounds	Prior knowledge	Sampling ranges	Number of samples	Number of valid samples
1	Sensitivity signs	$\mathfrak{X}_{i,1} = \mathfrak{X}^{sa}_{i,pos}(\Delta x^{sa}_i)$	250	0
2	Sensitivity signs	$\mathfrak{X}_{i,2} = \mathfrak{X}^{sa}_{i,pos}(2\Delta x^{sa}_i)$	1000	14
3	Preliminary feasibility	$\mathfrak{X}_{i,3} = \mathfrak{X}^{valid}_{i,2}$	500	253

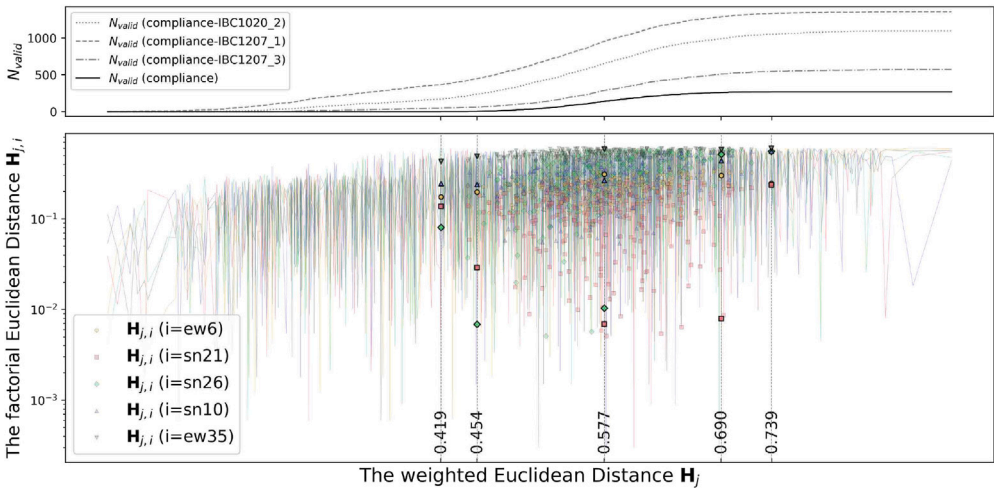


Fig. 11. Samples sorted via the weighted Euclidean Distance $\mathbf{H}_j$ to D_{ini}: the accumulated number of valid design alternatives N_{valid} that respectively comply with single building IBC rules and all considered IBC rules (upper), and the composition of $\mathbf{H}_j$ by factorial Euclidean Distances $\mathbf{H}_{j,i}$ for each variant (lower).

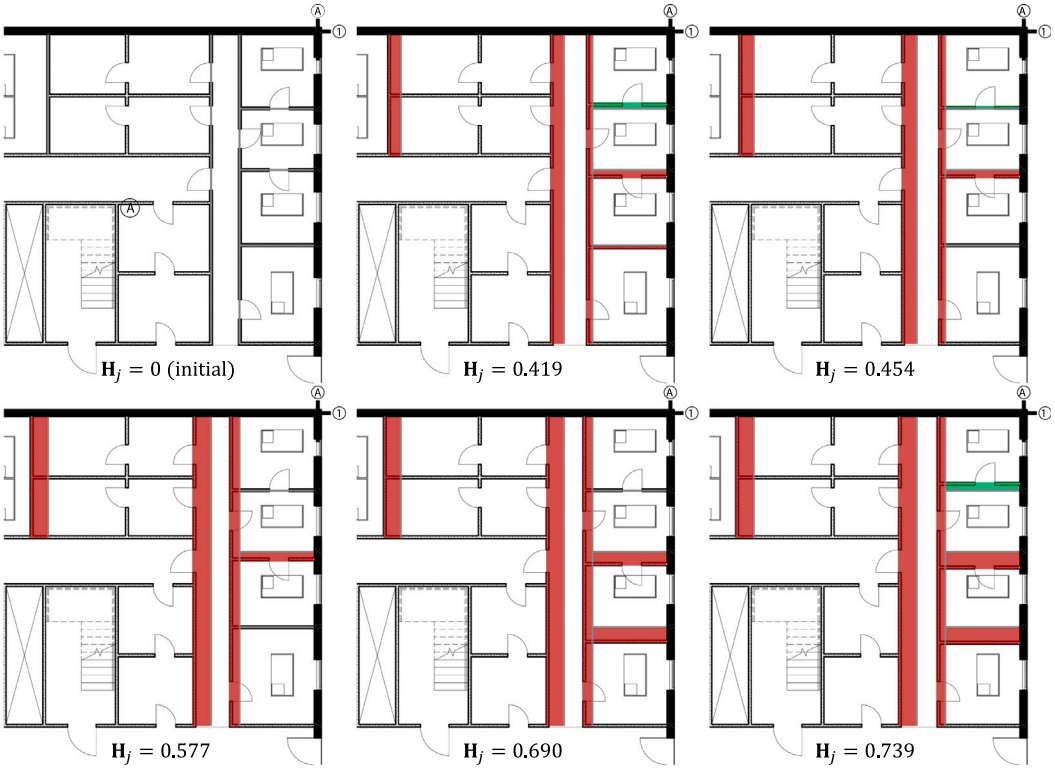


Fig. 12. Comparison of valid design alternatives across a diverse range of $\mathbf{H}_j$, from the identified smallest to largest values. Initial wall locations are shaded in grey, with red indicating wall movement in a further direction and green indicating movement in a closer direction, relative to reference grids 1 and A.

to individual building regulations and their ensemble. N_{valid} increases significantly within the range $\mathbf{H}_j = [0.419, 0.739]$, indicating the primary range within which the majority of achievable valid design alternatives are located. Additionally, we examine the factorial Euclidean Distances $\mathbf{H}_{j,i}$ of valid alternatives to assess dissimilarities across design variables. A relatively small $\mathbf{H}_{j,i}$ indicates that only minor adjustments are made in specific design variables, helping to reveal potential model adaptations without altering those particular design variables.

Local details of the valid variants are selectively presented in Fig. 12, illustrating code-compliant design alternatives across a diverse range of $\mathbf{H}_j$ — from the smallest to largest values identified. The initial design is used as a reference to compare local variations, with

highlighted areas reflecting changes in the relevant design variables $\hat{X}$ between each valid alternative and the initial design. Based solely on $\mathbf{H}_j$, the design alternative that deviates the least from the initial design has $\mathbf{H}_j = 0.419$. As this variant involves the smallest total geometry changes, it can be considered the primary design alternative. More significant design changes are identified in valid variants with larger distances $\mathbf{H}_j \in \{0.690, 0.739\}$. Furthermore, several variants hold relatively small $\mathbf{H}_{j,i}$. For example, the variant with $\mathbf{H}_j = 0.454$ involves a minor change in the variable sn_{26}, while the variant with $\mathbf{H}j = 0.577$ shows relatively minor changes in both $sn26$ and sn_{21}.

The valid design alternatives were reviewed by the creator of the experimental model, who confirmed the appropriateness of these localized changes without needing to alter the core design principles. The designer approved these closest solutions, as they maintained design integrity while achieving code compliance. The alternative with $\mathbf{H}_j = 0.454$ met the requirements of IBC 1020.2, IBC 1207.1, and IBC 1207.3 with minimal geometric adjustments. Additionally, variants with minor factorial distances were also approved ($\mathbf{H}_j \in \{0.454, 0.577\}$), as they revealed design alternatives requiring adjustments in only a subset of the variables. This feedback supported finalizing the layout, showing that the design could effectively meet building requirements with localized adjustments.

5. Discussion

5.1. Contribution

The proposed Design Healing framework represents an initial approach to automating model adaptations for code compliance. This framework converts non-compliant initial designs into compliant alternatives, offering the following key contributions:

1. The framework bridges the gap by directly utilizing ACC results to correct design issues through an automated adaptation workflow, using formal representations of building models and ACC outcomes.
2. The topological propagation mechanism leverages graph representation to identify non-compliance-related elements, enabling dynamic localized changes while preserving original topological relationships.
3. The multi-step design space exploration utilizes SA and the cLHS sampling strategy to reduce dimensionality through prior knowledge, enabling efficient navigation toward feasible sub-spaces.
4. Building on initial design solutions from human experts, the Design Healing approach provides designer-centered support for achieving code-compliant alternatives. It respects the contextual knowledge and expertise of designers by updating constraints from the authoring side.

This paper introduces a framework for building design adaptation that achieves code compliance while preserving the original concept, automating post-checking design corrections for human designers. Sharing the design improvement goals of performance optimization studies, the framework distinguishes itself with a formal data representation of both the BIM model and ACC results. Leveraging the parametric nature of design modifications, design correction is transformed into an adaptable, formalized design space exploration process that manages design variable variations and evolving constraints from the authoring side. Validated by a real-world case study in architectural spatial design, the Design Healing framework demonstrates computational accessibility for advanced optimization techniques, with the capacity to address more complex rule-violation scenarios.

Taking the graph representation technique as a basis, the topological propagation algorithm identifies targeted design modifications by analyzing interrelationships among building components. Rather than applying design variations across the entire building, which can lead to high computational costs, the propagation algorithm resolves non-compliance by focusing on essential building components at dynamically expanding levels. It starts with localized adjustments and, through expanding propagation levels, incorporates additional, unrestricted elements as needed to achieve code compliance in surrounding areas. By integrating existing constraints – particularly non-negotiable hard constraints – into the propagation conditions, the algorithm demonstrates adaptability to diverse real-world design scenarios.

Achieving compliant designs that satisfy multiple interrelated regulations is challenging, as the design space expands significantly with each added variable. This complexity often forces designers into iterative cycles, where addressing one issue may inadvertently cause another requirement to be violated. SA has proven effective for narrowing variables in space exploration, with the potential for handling higher-dimensional spaces. However, a key aspect often overlooked in design optimization studies is the full utilization of SA outcomes, particularly the directional insights provided by Morris sensitivity indices. These indices offer directed influence information that can be instrumental in guiding the sampling of new design variants. The multi-step design space exploration approach leverages prior knowledge by combining SA and the cLHS sampling strategy to efficiently reduce dimensionality.

The Design Healing approach provides designer-centered support by building on initial design solutions crafted by human experts. The proposed framework achieves code-compliant alternatives by respecting designers' contextual knowledge and expertise, incorporating constraint updates from the authoring side as needed. When localized adjustments alone cannot yield compliant alternatives, designers can release additional constraints to explore broader design possibilities. Different from purely generative design methods, this approach does not require an exhaustive set of predefined design parameters in downstream stages, which may be impractical in real-world projects that have undergone ACC evaluations.

The term "healing" is used with a few focuses in the AEC industry. Some studies focus on geometric healing to fix issues like non-closed surfaces and errors in CAD models [73,74], while others apply semantic healing to enhance design knowledge, such as segment classification or improving BIM accuracy [75,76]. While previous research has focused on improving model consistency and quality, our approach emphasizes design correction for code compliance.

5.2. Limitations

Several limitations are acknowledged in this paper. First, a key assumption for our proposed framework is that the initial design conditions are appropriately oriented in the right direction by design experts. Based on this assumption, the Design Healing approach produces valid alternatives through local modifications without altering the original design topology. When feasible designs cannot be achieved for an adequately designed model, designers can intervene to expand the design space by adjusting assigned design constraints. This assumption is reasonable, especially for designs crafted by experienced experts. However, the design space may not yield valid solutions if the initial model significantly deviates from regulatory requirements, particularly regarding spatial and geometric issues. This framework does not account for the full design complexity in such cases and is unlikely to produce results if the model's baseline conditions are too far from compliance.

Secondly, this paper focuses exclusively on rules related to architectural spatial design without fully addressing the broader diversity and complexity of building regulations. Efficiency limitations may arise with rules requiring specific performance simulations, such as egress time for fire safety, as their time-intensive nature can reduce the effectiveness of the Design Healing framework. Addressing uninvestigated rules may require adjustments to the approach, including adaptations to graph structures to better capture relevant design dependencies. For

example, improving accessibility and evacuation may involve adding doors or stairways, making it necessary to embed accessibility relationships within the graph structure and propagation algorithm. Additionally, adapting the approach's representation parameters may be necessary to address requirements with differing compliance-checking logic. For instance, the reduction coefficient may be unnecessary for rules executed only once per model. Such differences could also impact the calculation of the weighted Euclidean Distance, which currently relies solely on topological connectivity as weights for comparing design dissimilarity. Distance calculations based on quantitative parameters may overlook the complexity of design changes required for specific structural and functional requirements. A more comprehensive measure of design dissimilarity is therefore required.

Finally, this paper examines design constraints within a limited scope. All potential restrictive requirements not evaluated by ACC are considered design constraints. Human designers must manually review and update these constraints when the framework fails to generate valid alternatives for building rules. However, conflicting constraints at various levels may arise from the authoring side, potentially reducing the approach's effectiveness. Currently, these constraints are not differentiated into structured categories, such as non-negotiable hard constraints versus flexible soft constraints. This limits the framework's ability to explore acceptable alternatives in a fully automated way. Therefore, further investigation is needed to incorporate classified design constraints to enable more intelligent and adaptable adjustments.

6. Conclusion

Although the ACC research has gained continuous popularity, post-checking design adaptation for code compliance remains underexplored. One key potential lies in using ACC results to correct the designs. We argue that the most efficient approach begins with the initial, potentially non-compliant design created by human experts, applying localized adjustments to generate code-compliant variants for designers. The proposed Design Healing framework formalizes engineering knowledge from building design data and ACC results to resolve code violations. Built on predefined parametric models, non-compliance-related design variables are identified through a propagation algorithm that leverages graph structures to progressively account for dependencies among building components and assigned constraints. Using these essential design variables, the prior knowledge-informed design space exploration efficiently adapts the design to valid alternatives through localized changes while preserving the topology originally constructed by designers.

The proposed framework incorporates constraints derived from designers' engineering knowledge and domain expertise, allowing for their adjustment to enable flexible human intervention when localized modifications are insufficient. This approach provides valuable support for building design professionals, researchers, and developers focused on design optimization, with promising potential to refine designs based on ACC results. Additionally, demonstrating the applicability of ACC for post-checking design correction highlights the need for standardized checking outputs. It indicates the importance of advancing ACC approaches to produce richer, computationally accessible results for seamless integration into automated design workflows.

Future research will first extend the framework to multi-disciplinary scenarios beyond architecture, supporting major, non-local design interventions and enabling broader modifications across the model. By integrating perspectives from structural engineering, mechanical systems, and environmental performance, the framework can address complex interactions and dependencies inherent in holistic building design. Consequently, adopting a more generic graph representation, such as a hierarchical graph, is necessary to capture diverse design interrelationships, enhancing the framework's ability to effectively model and manage complex scenarios. Secondly, developing an intelligent system to represent complex and diverse constraints in building design is essential, as these often involve intricate relationships among components and require enhanced adaptability. By enabling dynamic updates, such a system would allow designers to apply custom constraints as projects evolve, particularly when achieving code compliance is challenging. Classifying constraints into non-negotiable hard constraints and flexible soft constraints would further support this adaptability. Additionally, we acknowledge that the model's parametrics must be pre-defined. Therefore, we are investigating the automated parametrization of existing non-editable building models that lack predefined design parameters and necessary dependencies. We aim to broaden the framework's practical application to raw BIM models by enabling parametrization in non-parametric models.

CRediT authorship contribution statement

Jiabin Wu: Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Stavros Nousias:** Writing – review & editing, Supervision, Project administration. **André Borrmann:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

Declaration of competing interest

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

Acknowledgments

This research was funded by the TUM International Graduate School of Science and Engineering (IGSSE). The authors would like to extend our sincere appreciation to Rafael Sacks, Jimmy Abualdenien, and Rohit K. Dubey for their support during this research.

Appendix A. The topological propagation algorithm

Algorithm 1 The topological propagation algorithm for finding relevant building objects to overcome non-compliance

Input: $V_{ini}, C, G, \mathcal{T}, l_{max}$
Output: V_u, V_c
$l \leftarrow 1$ ▷ propagation level
$V_u \leftarrow V_{ini}$ ▷ propagated nodes
$V_c \leftarrow \emptyset$ ▷ constrained nodes
$V_{current} \leftarrow V_u$ ▷ starting nodes
while $l < l_{max}$ **do**
  $V_{new} \leftarrow \emptyset$
  for V in $V_{current}$ **do**
    $V_{new} \leftarrow V_{new} \cup \mathcal{T}(G(V, E), C)$
    if $V \cap C = \emptyset$ **then** ▷ unconstrained case
      $V_u \leftarrow V_u \cup V_{new}$
    else ▷ constrained case
      $V_c \leftarrow V_c \cup V_{new}$
    end if
  end for
  $V_{current} \leftarrow V_{new}$ ▷ update the starting nodes for the next level
  $l \leftarrow l + 1$
end while

Appendix B. The morris screening-based method

The Morris method evaluates the influence of the individual input variables using an elementary effect (EE_i) for each variable (Eq. (B.1)). The input set consists of i_{max} input variables.

$$EE_i = \frac{f(x_1, \ldots, x_i + \Delta, \ldots, x_{i_{max}}) - f(x_1, \ldots, i_{max})}{\Delta} \tag{B.1}$$

Following a standard SA practice, these variables are sampled with uniform input distributions and transformed from the unit hypercube to their actual values. Within the input variables' space, each model input is varied across p selected levels with an equally divided distance Δ between them. Consequently, the input space forms an i_{max} dimensional p level grid with pi_{max} points. The Morris SA sampling follows t_{max} different trajectories. Each trajectory initiates randomly over a uniform grid, and the subsequent points are obtained by changing one variable at a time [68]. This provides t_{max} estimates of the elementary effects for each input variable, resulting in a total of $t_{max}(i_{max}+1)$ samples. An optimization strategy for sampling trajectories can be used to maximize their dispersion in the input space [69,77,78]. This involves generating a high number of trajectories and subsequently selecting a subset of optimized trajectories based on the distance between them. In addition, μ_i and σ_i are two sensitivity measures that respectively represent the mean and the standard deviation of the distribution of the influence caused by x_i (Eq. (B.2)).

$$\mu_i = \frac{1}{t_{max}} \sum_{t=1}^{t_{max}} EE_{i,t}, \quad \sigma_i = \sqrt{\frac{1}{t_{max}-1} \sum_{t=1}^{t_{max}} (EE_{i,t} - \mu_i)^2} \tag{B.2}$$

Data availability

Data will be made available on request.

References

[1] C. Langenhan, M. Weber, M. Liwicki, F. Petzold, A. Dengel, Graph-based retrieval of building information models for supporting the early design stages, Adv. Eng. Inform. 27 (4) (2013) 413–426, http://dx.doi.org/10.1016/j.aei.2013.04.005.

[2] H. Mattern, M. König, BIM-based modeling and management of design options at early planning phases, Adv. Eng. Inform. 38 (August) (2018) 316–329, http://dx.doi.org/10.1016/j.aei.2018.08.007.

[3] A. Borrmann, M. König, C. Koch, J. Beetz, Building information modeling: Why? What? How? in: A. Borrmann, M. König, C. Koch, J. Beetz (Eds.), Building Information Modeling - Technology Foundations and Industry Practice, vol. 1, Springer, 2018, pp. 1–24, http://dx.doi.org/10.1007/978-3-319-92862-3_1.

[4] R. Sacks, C. Eastman, G. Lee, P. Teicholz, Core technologies and software, in: BIM Handbook, John Wiley & Sons, Ltd, 2018, pp. 32–84, http://dx.doi.org/10.1002/9781119287568.CH2.

[5] C. Eastman, P. Teicholz, R. Sack, K. Liston, BIM Handbook, A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors, Second Edition, John Wiley & Sons, Inc, Hoboken, 2011, ISBN: 9780470541371.

[6] R. Amor, J. Dimyadi, The promise of automated compliance checking, Dev. the Built Environ. 5 (October 2020) (2021) 100039, http://dx.doi.org/10.1016/j.dibe.2020.100039.

[7] C. Eastman, J. min Lee, Y. suk Jeong, J. kook 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.

[8] C. Preidel, A. Borrmann, BIM-based code compliance checking, in: A. Borrmann, M. König, C. Koch, J. Beetz (Eds.), Building Information Modeling - Technology Foundations and Industry Practice, vol. 1, Springer, 2018, pp. 367–381, http://dx.doi.org/10.1007/978-3-319-92862-3_22.

[9] Solibri Inc, 2023, https://www.solibri.com/solibri-office, Accessed: 2024-01-16.

[10] J. Wu, R.K. Dubey, J. Abualdenien, B. André, Model Healing: Toward a framework for building designs to achieve code compliance, in: Proc. of European Conference on Product and Process Modeling, CRC Press, 2022, pp. 450–457, http://dx.doi.org/10.1201/9781003354222-58.

[11] P.-c. Lee, T.-p. Lo, M.-y. Tian, D. Long, An efficient design support system based on automatic rule checking and case-based reasoning, Constr. Manag. 23 (2019) 1952–1962, http://dx.doi.org/10.1007/s12205-019-1750-2.

[12] C. Sydora, E. Stroulia, Rule-based compliance checking and generative design for building interiors using BIM, Autom. Constr. 120 (July) (2020) 103368, http://dx.doi.org/10.1016/j.autcon.2020.103368.

[13] H. Liu, J.C. Cheng, V.J. Gan, S. Zhou, A knowledge model-based BIM framework for automatic code-compliant quantity take-off, Autom. Constr. 133 (October 2021) (2022) 104024, http://dx.doi.org/10.1016/j.autcon.2021.104024.

[14] International Code Council (ICC), International building code (IBC 2018), 2021, URL https://codes.iccsafe.org/content/IBC2018P6.

[15] A.S. Ismail, K.N. Ali, N.A. Iahad, A review on BIM-based automated code compliance checking system, in: International Conference on Research and Innovation in Information Systems, ICRIIS, IEEE, 2017, http://dx.doi.org/10.1109/ICRIIS.2017.8002486.

[16] J. Zhang, N.M. El-Gohary, Integrating semantic NLP and logic reasoning into a unified system for fully-automated code checking, Autom. Constr. 73 (2017) 45–57, http://dx.doi.org/10.1016/j.autcon.2016.08.027.

[17] X. Wang, N. El-Gohary, Deep learning-based relation extraction and knowledge graph-based representation of construction safety requirements, Autom. Constr. 147 (June 2022) (2023) 104696, http://dx.doi.org/10.1016/j.autcon.2022.104696.

[18] F. Yang, J. Zhang, Prompt-based automation of building code information transformation for compliance checking, Autom. Constr. 168 (PA) (2024) 105817, http://dx.doi.org/10.1016/j.autcon.2024.105817.

[19] T. Bloch, A. Borrmann, P. Pauwels, Graph-based learning for automated code checking – Exploring the application of graph neural networks for design review, Adv. Eng. Inform. 58 (August) (2023) http://dx.doi.org/10.1016/j.aei.2023.102137.

[20] P. Pauwels, T.M. de Farias, C. Zhang, A. Roxin, J. Beetz, J. De Roo, C. Nicolle, A performance benchmark over semantic rule checking approaches in construction industry, Adv. Eng. Inform. 33 (2017) 68–88, http://dx.doi.org/10.1016/j.aei.2017.05.001.

[21] C. Preidel, A. Borrmann, Towards code compliance checking on the basis of a visual programming language, J. Inf. Technol. Constr. 21 (July) (2016) 402–421, URL https://www.itcon.org/2016/25.

[22] M. Häußler, S. Esser, A. Borrmann, Code compliance checking of railway designs by integrating BIM, BPMN and DMN, Autom. Constr. 121 (2021) http://dx.doi.org/10.1016/j.autcon.2020.103427.

[23] J. Soliman-Junior, P. Tzortzopoulos, M. Kagioglou, Designers' perspective on the use of automation to support regulatory compliance in healthcare building projects, Constr. Manag. Econ. 40 (2) (2022) 123–141, http://dx.doi.org/10.1080/01446193.2021.2022176.

[24] P.C. Lee, D. Long, B. Ye, T.P. Lo, Dynamic BIM component recommendation method based on probabilistic matrix factorization and grey model, Adv. Eng. Inform. 43 (May 2019) (2020) http://dx.doi.org/10.1016/j.aei.2019.101024.

[25] Y.W. Zhou, Z.Z. Hu, J.R. Lin, J.P. Zhang, A review on 3D spatial data analytics for building information models, Arch. Comput. Methods Eng. 27 (5) (2020) 1449–1463, http://dx.doi.org/10.1007/s11831-019-09356-6.

[26] X. Zhao, L. Huang, Z. Sun, X. Fan, M. Zhang, Compliance checking on topological spatial relationships of building elements based on building information models and ontology, Sustain. (Switzerland) 15 (14) (2023) http://dx.doi.org/10.3390/su151410901.

[27] C. Eastman, D.S. Parker, T.-s. Jeng, Managing the integrity of design data generated by multiple applications: The principle of patching, Res. Eng. Des. 9 (3) (1997) 125–145, http://dx.doi.org/10.1007/BF01596599.

[28] S. Esser, S. Vilgertshofer, A. Borrmann, Graph-based version control for asynchronous BIM collaboration, Adv. Eng. Inform. 53 (June) (2022) 101664, http://dx.doi.org/10.1016/j.aei.2022.101664.

[29] S. Isaac, F. Sadeghpour, R. Navon, Analyzing building information using graph theory, in: ISARC 2013 - 30th International Symposium on Automation and Robotics in Construction and Mining, Held in Conjunction with the 23rd World Mining Congress, (August 2013) 2013, pp. 1013–1020, http://dx.doi.org/10.22260/isarc2013/0111.

[30] A. Khalili, D.K.H. Chua, IFC-based graph data model for topological queries on building elements, J. Comput. Civ. Eng. 29 (3) (2015) 04014046, http://dx.doi.org/10.1061/(asce)cp.1943-5487.0000331.

[31] V. Donato, Towards design process validation integrating graph theory into BIM, Archit. Eng. Des. Manag. 13 (1) (2017) 22–38, http://dx.doi.org/10.1080/17452007.2016.1208602.

[32] W. Solihin, C. Eastman, A knowledge representation approach in BIM rule requirement analysis using the conceptual graph, J. Inf. Technol. Constr. 21 (November) (2016) 370–402, URL http://www.itcon.org/2016/24.

[33] Y. Zhou, S.M. Asce, W. Solihin, J.K.W. Yeoh, A.M. Asce, Facilitating knowledge transfer during code compliance checking using conceptual graphs, J. Comput. Civ. Eng. 37 (5) (2023) http://dx.doi.org/10.1061/JCCEE5.CPENG-4884.

[34] X. Xu, H. Cai, Semantic approach to compliance checking of underground utilities, Autom. Constr. 109 (May 2019) (2020) 103006, http://dx.doi.org/10.1016/j.autcon.2019.103006.

[35] J. Peng, X. Liu, Automated code compliance checking research based on BIM and knowledge graph, Sci. Rep. 13 (1) (2023) 1–12, http://dx.doi.org/10.1038/s41598-023-34342-1.

[36] G. Lee, R. Sacks, C.M. Eastman, Specifying parametric building object behavior (BOB) for a building information modeling system, Autom. Constr. 15 (6) (2006) 758–776, http://dx.doi.org/10.1016/j.autcon.2005.09.009.
[37] R. Sacks, C.M. Eastman, G. Lee, Parametric 3D modeling in building construction with examples from precast concrete, Autom. Constr. 13 (3) (2004) 291–312, http://dx.doi.org/10.1016/S0926-5805(03)00043-8.
[38] J.E. Harding, P. Shepherd, Meta-parametric design, Des. Stud. 52 (2017) 73–95, http://dx.doi.org/10.1016/j.destud.2016.09.005.
[39] S. Krish, A practical generative design method, Comput. Aided Des. 43 (1) (2011) 88–100, http://dx.doi.org/10.1016/j.cad.2010.09.009.
[40] A. Tedeschi, F. Wirz, AAD Algorithms-Aided Design: Parametric Strategies using Grasshopper, Edizioni Le Penseur, 2014, ISBN: 9788895315300.
[41] P. Sanguinetti, S. Abdelmohsen, J. Lee, J. Lee, H. Sheward, C. Eastman, General system architecture for BIM: An integrated approach for design and analysis, Adv. Eng. Inform. 26 (2) (2012) 317–333, http://dx.doi.org/10.1016/j.aei.2011.12.001.
[42] D. Aupy, Generative design for hospital pharmacy: optimizing spaces /flows with dynamo/refinery, 2023, URL https://www.autodesk.com/autodesk-university/class/Generative-design-Hospital-Pharmacy-optimizing-spacesflows-DynamoRefinery-2019.
[43] M. Marinov, M. Amagliani, T. Barback, J. Flower, S. Barley, S. Furuta, P. Charrot, I. Henley, N. Santhanam, G.T. Finnigan, S. Meshkat, J. Hallet, M. Sapun, P. Wolski, Generative design conversion to editable and watertight boundary representation, Comput. Aided Des. 115 (2019) 194–205, http://dx.doi.org/10.1016/j.cad.2019.05.016.
[44] V. Granadeiro, L. Pina, J.P. Duarte, J.R. Correia, V.M. Leal, A general indirect representation for optimization of generative design systems by genetic algorithms: Application to a shape grammar-based design system, Autom. Constr. 35 (2013) 374–382, http://dx.doi.org/10.1016/j.autcon.2013.05.012.
[45] L. Graff, H. Harbrecht, M. Zimmermann, On the computation of solution spaces in high dimensions, Struct. Multidiscip. Optim. 54 (4) (2016) 811–829, http://dx.doi.org/10.1007/s00158-016-1454-x.
[46] M. Zimmermann, J.E. von Hoessle, Computing solution spaces for robust design, Internat. J. Numer. Methods Engrg. 94 (3) (2013) 290–307, http://dx.doi.org/10.1002/nme.4450.
[47] O. Gassmann, M. Zeschky, Opening up the solution space: The role of analogical thinking for breakthrough product innovation, Creativity Innov. Manag. 17 (2) (2008) 97–106, http://dx.doi.org/10.1111/j.1467-8691.2008.00475.x.
[48] N.C. Brown, C.T. Mueller, Automated performance-based design space simplification for parametric structural design, in: Proceedings of the IASS Annual Symposium 2017, 2017, URL http://digitalstructures.mit.edu/files/2017-11/iass-17-paper-ncb-revision.pdf.
[49] R. Danhaive, C.T. Mueller, Design subspace learning: Structural design space exploration using performance-conditioned generative modeling, Autom. Constr. 127 (March) (2021) 103664, http://dx.doi.org/10.1016/j.autcon.2021.103664.
[50] D.J. Gerber, S.H. Lin, B. Pan, A.S. Solmaz, Design optioneering: Multi-disciplinary design optimization through parameterization, domain integration and automation of a genetic algorithm, in: Proceedings of the 2012 Symposium on Simulation for Architecture and Urban Design, Vol. 44, (8 BOOK) 2012, pp. 33–40, URL https://dl.acm.org/doi/10.5555/2339453.2339464.
[51] N.C. Brown, C.T. Mueller, Design variable analysis and generation for performance-based parametric modeling in architecture, Int. J. Archit. Comput. 17 (1) (2019) 36–52, http://dx.doi.org/10.1177/1478077118799491.
[52] M.M. Singh, C. Deb, P. Geyer, Early-stage design support combining machine learning and building information modelling, Autom. Constr. 136 (February) (2022) 104147, http://dx.doi.org/10.1016/j.autcon.2022.104147.
[53] J. Ortiz, J. Summers, J. Coykendall, T. Roberts, R. Rai, A topological formalism for quantitative analysis of design spaces, in: Proceedings of the Design Society, Vol. 1, (August) 2021, pp. 293–302, http://dx.doi.org/10.1017/pds.2021.30.
[54] E. Kang, E. Jackson, W. Schulte, An approach for effective design space exploration, in: Foundations of Computer Software. Modeling, Development, and Verification of Adaptive Systems, 2010, pp. 33–54, http://dx.doi.org/10.1007/978-3-642-21292-5_3.
[55] J. Goffart, M. Woloszyn, EASI RBD-FAST: An efficient method of global sensitivity analysis for present and future challenges in building performance simulation, J. Build. Eng. 43 (July) (2021) 103129, http://dx.doi.org/10.1016/j.jobe.2021.103129.
[56] E. Borgonovo, E. Plischke, Sensitivity analysis: A review of recent advances, European J. Oper. Res. 248 (2016) 869–887, http://dx.doi.org/10.1016/j.ejor.2015.06.032.
[57] C. Daniel, One-at-a-time plans, J. Amer. Statist. Assoc. 68 (342) (1973) 353–360, http://dx.doi.org/10.1080/01621459.1973.10482433.
[58] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, S. Tarantola, Variance-based methods, in: Global Sensitivity Analysis. the Primer, 2007, pp. 155–182, http://dx.doi.org/10.1002/9780470725184.ch4.
[59] B. Iooss, P. Lemaître, A review on global sensitivity analysis methods, Oper. Research/ Comput. Sci. Interfaces Ser. 59 (2015) 101–122, http://dx.doi.org/10.1007/978-1-4899-7547-8_5.
[60] D. Douglas-Smith, T. Iwanaga, B.F. Croke, A.J. Jakeman, Certain trends in uncertainty and sensitivity analysis: An overview of software tools and techniques, Environ. Model. Softw. 124 (September 2019) (2020) 104588, http://dx.doi.org/10.1016/j.envsoft.2019.104588.
[61] M.M. Singh, P. Geyer, Information requirements for multi-level-of-development BIM using sensitivity analysis for energy performance, Adv. Eng. Inform. 43 (September 2019) (2020) 101026, http://dx.doi.org/10.1016/j.aei.2019.101026.
[62] I. Sobol, Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates, Math. Comput. Simulation 55 (2001) 271–280, http://dx.doi.org/10.1016/S0378-4754(00)00270-6.
[63] I.M. Sobol', S. Kucherenko, A new derivative based importance criterion for groups of variables and its link with the global sensitivity indices, Comput. Phys. Comm. 181 (7) (2010) 1212–1217, http://dx.doi.org/10.1016/j.cpc.2010.03.006.
[64] Z. Pang, Z. O'Neill, Y. Li, F. Niu, The role of sensitivity analysis in the building performance analysis: A critical review, in: Energy and Buildings, Vol. 209, Elsevier, 2020, 109659, http://dx.doi.org/10.1016/j.enbuild.2019.109659.
[65] A.B. Owen, On dropping the first sobol' point, Springer Proc. Math. Stat. 387 (Mc) (2022) 71–86, http://dx.doi.org/10.1007/978-3-030-98319-2_4.
[66] L. Paleari, E. Movedi, M. Zoli, A. Burato, I. Cecconi, J. Errahouly, E. Pecollo, C. Sorvillo, R. Confalonieri, Sensitivity analysis using Morris: Just screening or an effective ranking method? Ecol. Model. 455 (January) (2021) 109648, http://dx.doi.org/10.1016/j.ecolmodel.2021.109648.
[67] J. Guo, M. Li, Y. Jin, C. Shi, Z. Wang, Energy prediction and optimization based on sequential global sensitivity analysis: The case study of courtyard-style dwellings in cold Regions of China, Buildings 12 (8) (2022) 1132, http://dx.doi.org/10.3390/buildings12081132.
[68] M.D. Morris, Factorial sampling plans for preliminary computational experiments, Technometrics 33 (May) (1991) 161–174, http://dx.doi.org/10.2307/1269043.
[69] F. Campolongo, J. Cariboni, A. Saltelli, An effective screening design for sensitivity analysis of large models, Environ. Model. Softw. 22 (10) (2007) 1509–1518, http://dx.doi.org/10.1016/j.envsoft.2006.10.004.
[70] M.D. McKay, R.J. Beckman, W.J. Conover, Comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 21 (2) (1979) 239–245, http://dx.doi.org/10.1080/00401706.1979.10489755.
[71] D. Autodesk, 2021, https://dynamobim.org/, Accessed: 2023-08-30.
[72] RIBA, RIBA Plan of Work 2020, Royal Institute of British Architects – RIBA, London, UK, 2021, URL https://www.architecture.com/knowledge-and-resources/resources-landing-page/riba-plan-of-work.
[73] J. Yang, S. Han, Repairing CAD model errors based on the design history, Comput. Aided Des. 38 (6) (2006) 627–640, http://dx.doi.org/10.1016/j.cad.2006.02.007.
[74] B. Wassermann, S. Kollmannsberger, S. Yin, L. Kudela, E. Rank, Integrating CAD and numerical analysis: 'Dirty geometry' handling using the Finite Cell Method, Comput. Methods Appl. Mech. Engrg. 351 (2019) 808–835, http://dx.doi.org/10.1016/j.cma.2019.04.017.
[75] F.C. Collins, M. Ringsquandl, A. Braun, D.M. Hall, A. Borrmann, Shape encoding for semantic healing of design models and knowledge transfer to Scan-to-BIM, Proc. Inst. Civ. Eng. - Smart Infrastruct. Constr. 175 (2022) 1–21, http://dx.doi.org/10.1680/jsmic.21.00032.
[76] K. Forth, J. Abualdenien, A. Borrmann, Calculation of embodied GHG emissions in early building design stages using BIM and NLP-based semantic model healing, Energy Build. 284 (2023) 112837, http://dx.doi.org/10.1016/j.enbuild.2023.112837.
[77] F. Campolongo, A. Saltelli, J. Cariboni, From screening to quantitative sensitivity analysis. A unified approach, Comput. Phys. Comm. 182 (4) (2011) 978–988, http://dx.doi.org/10.1016/j.cpc.2010.12.039.
[78] M.V. Ruano, J. Ribes, A. Seco, J. Ferrer, An improved sampling strategy based on trajectory design for application of the Morris method to systems with many input factors, Environ. Model. Softw. 37 (2012) 103–109, http://dx.doi.org/10.1016/j.envsoft.2012.03.008.