



Automated rule-based safety inspection and compliance checking of temporary guardrail systems in construction

K.W. Johansen^a, J. Teizer^{a,*}, C. Schultz^b

^a Department of Civil and Mechanical Engineering, Technical University of Denmark, Denmark

^b Department of Electrical and Computer Engineering, Aarhus University, Denmark

ARTICLE INFO

Keywords:

Automated construction safety inspection
Construction safety ontology
Continuous inspection
Digital safety rule-checking and reasoning
Fall protection systems
Rule-based inference
Photogrammetry
Point cloud data
Human-assisted decision-making software tools

ABSTRACT

The construction industry records more hazards compared to any other sector. Protective equipment, such as guardrail systems, is essential for protecting workers from deadly falls but may quickly become noncompliant after installation. Yet, many construction projects do not have the resources to dedicate personnel to perform the inspection as frequently as needed. Therefore, this paper proposes an automated rule-based inspection and compliance-checking system that can assist the responsible personnel in detecting faulty guardrails in live work environments. The classification approach utilizes safety design and mimics the steps of human guardrailing compliance assessment, which enforces simplicity and transparency, allowing the human domain expert to remain in control. Even under scarce data availability, this first-of-a-kind classification approach is reliable and scalable and successfully classifies 21 predefined and 9 validation scenarios of guardrail systems for fall protection.

1. Introduction

The construction industry has abysmal records regarding fatal hazards compared to other sectors. One of the factors playing a significant role in these records is the dynamic environment in which the construction process is carried out. There are no two identical construction sites, and barely two days of production are performed in identical situations as the building state progresses. Therefore, the location and environment of tasks are constantly changing [1]. The ever-changing environment makes it hard to keep the safety plan up to date, but it is even harder to keep installed hazard mitigation equipment compliant with the safety plan and regulations. Tremendous effort is allocated to safety planning and inspection, but the manual processes are subject to errors for several reasons, e.g., fatigue of repetition, unplanned removal of safety equipment to perform work, lack of reinstallation by workers, and fast addition of new potentially hazardous areas. In fact, 38 % of all work-related fatalities in the U.S. stem from falls from height hazards, which makes it the deadliest hazard type in construction [2]. The preferred protection equipment for fall hazards is guardrails, but those need frequent inspections to ensure safe work environments. This work investigates how an automated inspection procedure that can cope with the dynamic environment of a construction site can facilitate the

identification of safety mitigation equipment compliance issues, enabling the safety-responsible person to focus on correcting identified issues instead of spending as much time searching for them.

While the safety compliance checking task is labor-intensive and time-consuming, it is a critical task that can potentially prevent accidents in the construction industry and, thus, cannot be omitted [3]. This also means that solutions that seek to automate the task must be highly reliable. A reliable solution includes properties such as being transparent, completeness-assessable, explainable, extensible, and pessimistic (we further explain the meaning of these properties in our methodology Section 3). Additionally, the approach must be reliable even when sensor data about the construction site is scarce. Collecting data about hazardous situations in construction is a complex and challenging task, as these situations cannot persist until data has been captured (i.e., such hazardous situations must be made safe as soon as possible to maximize worker safety).

Example: An active construction site, e.g., a high-rise building currently being built, that contains *falls from height* hazards. The construction project Building Information Modeling (BIM) model assembles the hazards and their required mitigation measures, such as guardrails, are modeled. Utilizing the safety-assessed BIM, a new safety analysis tool that takes sensor data input from the construction site (e.g., video or

* Corresponding author.

E-mail address: teizerj@dtu.dk (J. Teizer).

<https://doi.org/10.1016/j.autcon.2024.105849>

Received 11 April 2024; Received in revised form 9 October 2024; Accepted 24 October 2024

Available online 20 November 2024

0926-5805/© 2024 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

images captured by a drone) to perform compliance checking and highlights hazardous locations in red in a BIM model viewer. The areas should be highlighted when the mitigation measures do not comply with safety regulations. Moreover, the safety analysis tool would associate these uncompliant locations with a description of the violation to let the safety inspector know exactly what to bring to repair or replace the faulty pieces and maintain site safety before workers get hurt.

Identification and classification of objects often rely on computer vision and machine learning [4]. The functionality of both approaches is based on a training procedure, where the neurons in the network are optimized to predict a class based on the similarity to instances in a labeled training data set. This means that training a neural network to classify instances of guardrails requires a good representation of each violation with several instances, i.e., data that captures the violation.

An alternative approach is rule-based, which we investigate in this study. The rule-based approach is initiated by formalizing the properties of the guardrails into logic facts and rules. A set of facts and rules is referred to as a knowledge base, and a reasoning engine is used to infer new facts that are logically consistent with the knowledge base. Different kinds of inferences are made using different kinds of rule-based systems, e.g., via query answering (e.g., Prolog [5]), (stable) model generation (e.g., Answer Set Programming [6]), rule execution (e.g., production systems [7]), rule-based constraint programming (e.g., Constraint Handling Rules [8]), and others. For example, the guardrail regulation specifies the number of required boards, and therefore, a guardrail would violate this rule if it does not meet the requirement. Formalizing the regulations as rules and encoding the guardrails on a particular construction site as facts makes it possible to find all guardrails that violate the regulation and categorize them as in compliant with the regulation.

In this work, we conduct a case study-based investigation to address our research questions. Specifically, we propose and develop a rule-based strategy for automatically classifying guardrails on construction sites. Our approach has the potential to enhance compliance by allowing safety-responsible individuals to focus on identified violations rather than non-violating installations. We explore practical methods for automating guardrail inspection, addressing research questions RQ1–RQ3. Through the investigation, we introduce a novel compliance classification strategy that considers real-world requirements and limitations. The contributions of our work, summarized as C1–C4, offer insights into a less explored yet highly effective approach.

Research questions:

- RQ 1: How can a declarative, knowledge-based approach be used to perform compliance checking of temporary guardrails in an automated approach under scarce data availability?
- RQ 2: How can a small set of predefined scenarios (e.g., 10–50) be sufficient to form a working compliance-checking approach to classify a set of validation scenarios captured around the construction scenario?
- RQ 3: Does the classification approach scale to be feasible in realistic full-size construction site scenarios consisting of hundreds or thousands of mitigation strategy instances (i.e., instances of the planned placement of guardrails, safety nets, cones, fences, etc.)?

Contributions:

- C 1: We present a new strategy to capture guardrailing conditions from point cloud data.
- C 2: We present and validate a novel, scalable, and reliable approach to perform compliance classification on temporary safety guardrails on construction sites where sensor data is scarce.
- C 3: We freely share the knowledge base for the predefined scenarios and the validation scenarios on GitHub for others to advance and compare.

- C 4: We freely share the synthetic knowledge base created for the feasibility study on GitHub.

2. Related work

2.1. Safety planning

BIM models comprise essential information about the building. This includes but is not limited to the geometry, which can be used in the assessment of rule compliance. Rule compliance covers many different areas, such as fire simulation [9], accessibility [10], and visible access [11]. BIM also plays a vital role in construction worker safety rule checking. Several researchers have been investigating the identification of hazards based on the BIM model and its element's geometries. Safety considerations can be included in the guidance of design choices that are based on formalized construction worker suggestions [12]. However, some of the hazards, specifically fall hazards, can also be mitigated directly based on the BIM element geometries [13–15]. The analysis is performed and disseminated using different approaches. Some use semantic approaches, where hazards are shared with the designer or safety-responsible person [16–19]. An alternative approach is to inject mitigations into the BIM model when hazards are identified [13,15,20–23]. This often means that the scope of hazard types is narrower and that the geometry and location of the mitigation equipment can be extracted directly from the BIM model. The digitally captured safety design can be used to facilitate the automatization of other safety tasks, e.g., inspection, as the knowledge about where and what kind of protection equipment to expect.

2.2. Progress monitoring

Safety equipment inspection is highly related to construction progress monitoring, which compares the as-built state to the as-planned state of a construction project [24]. Progress monitoring is performed in different resolutions, i.e., binary, stating if an object is completed or not, and percentage-wise, where an object can be considered partially done. The percentage-wise approach, depending on the tasks, often requires images to assess which tasks have been completed for an object, e.g., if a wall is built but not yet painted, which depends on the resolution of the expected output. For some tasks, though, it is possible to assess the percentage completion of a masonry wall based on a point cloud [25], but at the same time, this has been studied using RGB (red, green, blue colored) images and Convolutional Neural Networks (CNN) [26]. Most of the studies mentioned in [24] are based on point cloud data from Light Detection and Ranging (LiDAR), photogrammetry processed imagery data, or Simultaneous Localization and Mapping (SLAM). The data acquisition technique impacts the quality of the resulting point cloud and must be selected to correspond with the application's requirements. The overall topic is also referred to as Scan-to-BIM and has been investigated recently in [27], and in a robotic-assisted approach [28].

2.3. Safety equipment detection

Specifically for safety equipment detection, which is more related to this work, existing studies also include data collection and compliance checking of safety properties in the as-built environment. Similar to the overall progress monitoring, some studies are based on images, and others are based on point clouds from various sources. One example of a study based on two-dimensional (2D) images and CNN is [29]. The approach utilizes transfer learning, allowing the authors to benefit from training performed on the network to solve another similar task. The (Visual Geometry Group)VGG-16 [30] model can be adopted and further trained specifically for guardrail detection. Their training dataset consisted of 4000 augmented images, and another 4000 evenly collected from real construction sites and Google was used to validate the model's performance. With the data set and the transfer learning strategy, the

model performance becomes 96.5 % for one type of manufactory-produced guardrails. [31] is another image-based study, but instead of regular images collected from the ground, the study is based on video recording from an Unmanned Aerial Vehicle (UAV). Key-frames are automatically identified from the video recording, and topology-based detection of guardrails posts is performed. The authors improve the detection through definitions of constraint where the relationship and distance between railing poles and the floor are constrained. Their approach is based on Machine Learning (ML) rather than Deep Learning (DL); thus, the training data set size requirements are lowered. While the data requirements are lowered, ML approaches still need large amounts of training data to represent the many ways safety equipment can be non-compliant. The training data is cumbersome to collect in live construction sites with a reasonable variety to make the network perform well, as it requires the existence of unsafe situations until data is collected.

2.4. Safety compliance checking approaches

Automated safety inspection, which is yet more relevant to this work, has been based on many different data sources and analyzed using different approaches. One way is to apply a DL approach on images to mask the railings and changes between subsequent frames [32]. The study utilizes a U-Net segmentation model [33], which allowed the training to be done on a relatively small data set compared to other DL approaches. In the study, 1560 images were used during training, and an additional 129 images were used for testing. All images were taken from the construction site's tower crane and labeled for model training and testing. The study also identifies where the railings are missing based on the previous frames, which means that the detection is limited to the lack of railings that have once been installed. To do so, the authors utilize and compare The Mask Comparison Approach (MCA) and the Missing Object Detection Approach (MODA), where the latter performs best in detecting where guardrails have been removed in the construction situation. While the results of this work look promising, it requires the line of sight from the crane video camera, which is often not possible to ensure.

Point cloud data (PCD) has been used for many applications in construction research, and guardrail inspection is no exception. One study presented a workflow in which individual guardrails can be inspected for regulation compliance [34]. The PCD was created from images collected with a UAV that were subsequently processed with Structure from Motion (SfM) [61]. The compliance checking approach is based on finding openings between guardrail boards and flooring in the PCD. Based on the regulation, the openings are classified and used to identify hazards in the collection environment. The investigation was performed under controlled circumstances in a lab, where the individual cases were introduced subsequently. Instead of creating the PCD from images in an SfM approach, it has also been investigated using a terrestrial laser scanner, which drastically increases the resolution of the acquired data [35]. Additionally, Mobile Laser Scanners (MLS) have been applied in the data collection to collect data for the safety assessment [36]. These studies surveyed scaffolding and their work platforms in an orientation-based stepwise approach, which (1) extracts the scaffold from the complete PCD, (2) extracts the work platforms, (3) identifies vertical railing components, and (4) identifies the horizontal railing components. Subsequently, each component is checked for compliance with the regulation. The results yielded promising potential. However, the authors state limitations regarding the complexity of scaffold shape and time-consuming manual data collection.

Besides collecting on-site data, studies have used other available information sources, such as the BIM model, which can be further facilitated by studies such as [15,16,37]. Using a tower crane-mounted camera in combination with the information in the BIM model for investigation of construction is, for example, done by [38]. Through matrix operation, the authors translated and rotated the images from the camera to the BIM model, which allows them to find objects and

locations in the video frames. The process enabled the identification of safety-related objects, such as planned hazard zones, and the dissemination of hazardous situations found in the video frames through the BIM model. The identification of hazards in the video is mainly focused on detecting human workers, unprotected, and protected openings. With these detections, it is possible to identify workers approaching unprotected openings, which can lead to hazardous situations. Including information from the BIM model relieves some responsibility from the detection algorithm but also requires the BIM model to hold additional information, which can add manual tasks to the safety-responsible person.

Automated compliance checking approaches in the literature have shown very promising results in different applications of construction safety equipment. However, many of these approaches also rely on large data sets for the algorithm or network training. The data sets are seldom shared and represent specific guarding components (e.g., boards with markers or specific paintings). While these resources are major contributions to the body of research, there is still room for improvement towards more generally applicable frameworks. Such frameworks should require less training data collection to make them easily applicable, utilize existing information to reduce the responsibility of the compliance checking task and enforce transparency to keep humans in control.

2.5. Rule-based approaches in construction

Rule-based approaches have been researched for around a half-century in construction, but mostly concerning automated construction code compliance checking [39]. There have been applications in all lifecycle phases of the construction industry (i.e., architecture, engineering, construction, operation and management, decommissioning, or repurposing). Compliance checking can be done by formalizing natural language rules and building model knowledge. The latter has been positively affected by the creation and promotion of OpenBIM as extraction and utilization of BIM data is easier across compliance-checking applications [40]. The usage of BIM, and especially OpenBIM, allows for, in addition to semantic information, geometry extraction and in combination with a spatial query language, it is possible to automate compliance checking of rules that rely on geometric properties [11,41].

As all rule-based approaches are initiated with formalizing the domain, much effort has been put into creating ontologies to facilitate the formalization procedure for different areas of code compliance checking applications [10,40,42–45]. They aim to formalize a specific domain regarding rules that apply, domain expert knowledge, and the information needed to check compliance (e.g., building objects in building design regulation studies). The normative representation of a rule usually ends up in the form *if x then y*, which can be implemented in most logic inference frameworks. The implementation is human-readable, and with proper implementation and documentation, it is possible to perform checking of instances in a domain while also accessing the detailed reasoning steps that lead to the inference. At the same time, it is possible to express the domain expert knowledge and even regional- or company-specific best practices, allowing the approach to perform the same checks as a human safety responsible person in a white-box strategy.

3. Methodology

This section describes how our research questions were approached to achieve the envisioned automated inspection and classification of safety mitigation equipment, such as temporary guardrails. The overall framework is illustrated in Fig. 1, which is split into three main steps: *planning, data collection and preparation, and classification*. The framework illustrates the full pipeline, but this study limits the scope due to the existing research [27,28,35,46,47,62,63] that already achieves as-built capture and registration with high-quality results. Due to the

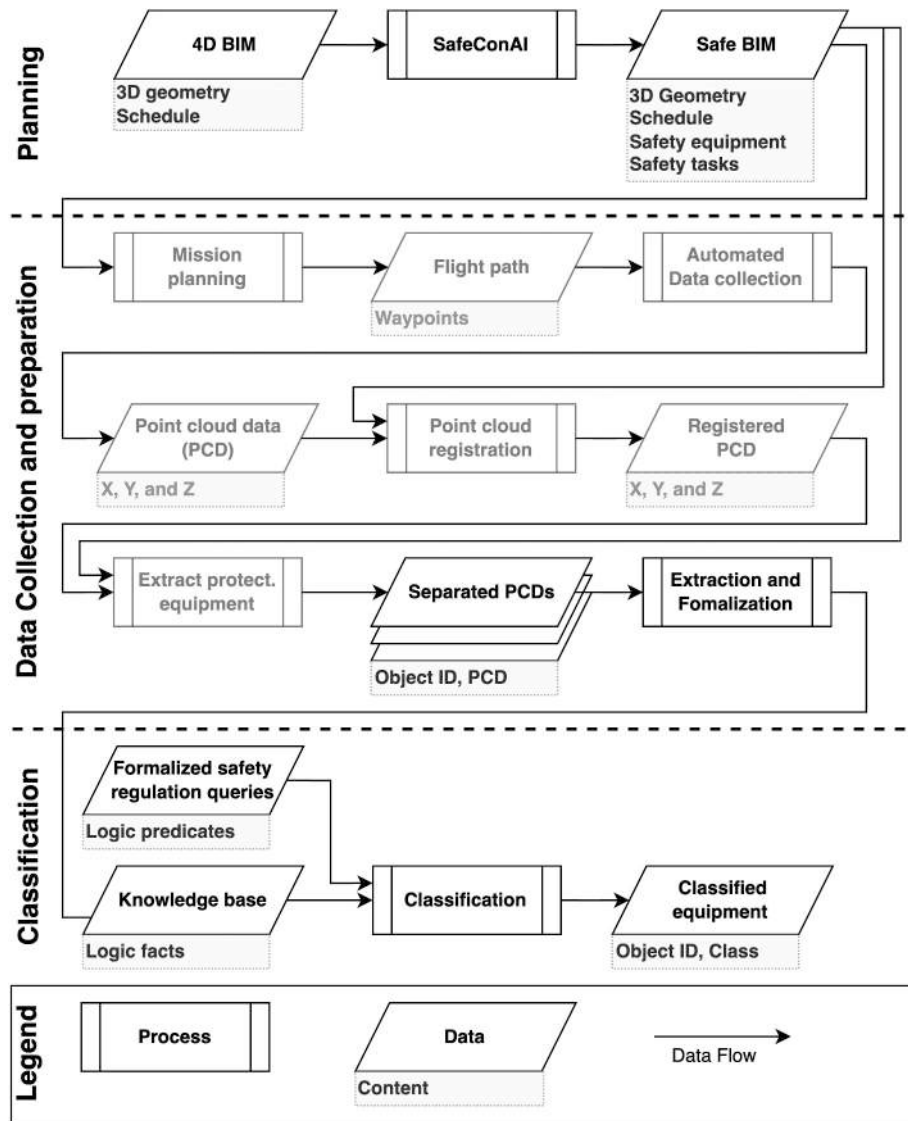


Fig. 1. Proposed framework for automated safety mitigation equipment inspection. The grayed boxes are assumed to exist and are not part of the concrete scope of this work.

existing work covering *mission planning*, *automated data collection*, *point cloud registration*, and *automated extraction* are not part of the scope of this work. Specifically in [46], The authors of this work also previously investigated an autonomous approach for mission planning, data collection, and point cloud registration, which, together with the results of the community, allow us to focus on the remaining part of the framework presented in Fig. 1.

3.1. Safety design and planning

The planning part of Fig. 1 refers to the process of creating a safe BIM model, which we assume to exist based on previous work by the authors (i. e., the application called *SafeConAI* [23]) but could also be based on other similar approaches [17,20,45,48–50]. *SafeConAI*'s result, which is called the *SafeBIM*, holds the original information from the incoming BIM model and the safety-related protective equipment, such as guardrails, cover panels, and safety nets. Based on the *SafeBIM*, it is known where to look for mitigation equipment, and the PCD can be cropped using a bounding box strategy. This is done with the location and geometry of the safety protection equipment, which is used to create the bounding box that is subsequently dilated so it also incorporates the surroundings of the protective equipment. The points from the PCD that fall inside the bounding box are

split into separate PCDs and labeled with the object's global unique identifier (GUID) from *SafeBIM*. This way, the link between the PCD and BIM object is maintained and can be utilized to disseminate the inspection results, e.g., compliance reporting through the BIM model, where the individual elements are colored based on their compliance.

3.2. Data collection and preparation

The proposed framework relies on PCD as the primary data source such that the collection can also happen when the construction site is less active or inactive as the PCD collecting devices do not rely on light (e.g., at night to avoid exposing the hazards that flying UAVs introduce [51]). As the PCD does not need daylight, a specialized UAV can autonomously collect the data without endangering workers, and the safety manager's fresh report of safety compliance is ready the next day before the work is continued. The UAV also means that the protection equipment can be accessed from outside the building, where obstructions are highly unexpected and unusual.

Integrating a digital safety plan to locate mitigation equipment and the selection of PCD enables the UAV to operate at night in an optimized pattern. Although continuously improving, given that battery life is a critical constraint for operating UAVs, it is essential for them to follow

the most efficient flight route. If the construction site and building are too large for a single UAV or flight to handle, the task can be distributed among multiple UAVs. The proposed framework identifies flights when no workers are present, for example during night or weekends, as the most viable solution for automated data collection. The risk of missing or faulty guardrails remains a concern during daytime operations when the construction site is typically very active. Only once the automated UAV flights are proven to be secure, safe, and stable, daytime operations should be considered. Proper regulations that yet need to be established can guide users. Such an approach would then also support the assessment of guardrails from the exterior, minimizing worker exposure to UAV presence and reducing the potential for collisions with other machinery and objects.

For this study, the broad scope of temporary construction safety equipment is narrowed to mainly focus on guardrailings, which are used frequently in the construction industry to prevent *falls from height* hazards. Temporary safety guardrails can be module-based or built in a particular situation, consisting of poles and wooden boards. The regulation on fall prevention is often clearly defined and straightforward for humans to check using a stepwise approach (e.g., German [52] or U.S. [53] regulation). The number of boards, poles per meter, minimum height, etc., are the things that a human would look for while performing the assessment.

3.3. Classification

As we require a transparent, explainable, extensible, and reliable classification method, we are basing this study on a logic inference approach that requires the mitigation equipment instance to be captured formally. The formalized knowledge consists of two main categories of expressions: boards and poles (i.e., collected data) and *SafeBIM*-related facts. The extraction utilizes a Hough transformation [54] to identify the prominent lines in a bitmap representation of the points in the bounding box cut PCD. With some additional preprocessing steps (shown in Fig. 4), the logic fact-representations of boards and poles are identified and collected in the formalized knowledge base (further explained in Section 4).

Table 1 explains the four pivotal operational properties of the proposed classification methodology: transparency, explainability, extensibility, and pessimism. These properties, initially introduced in the introductory section and further discussed in the literature review, are now explicated in the context of ML and rule-based approaches. In addition to these operational aspects, it is imperative to evaluate the creational attributes, such as community size and applicability, where ML approaches exhibit superiority. The ML paradigm, driven by data, discerns patterns within datasets, whereas the rule-based paradigm operates on explicitly formalized domain knowledge, necessitating manual encapsulation. This process of knowledge encapsulation lowers the need for data, relying instead on the meticulous articulation of domain expertise for computational interpretation. Moreover, rule-based systems do not inherently detect latent patterns in data, which could be instrumental in scenario identification; however, this also means they are devoid of inherent negative biases. Both paradigms present distinct advantages and limitations. Notwithstanding these considerations, the present study explores the rule-based approach's competencies within the guardrail classification realm.

Deciding on the approach used in the research presented in this paper came down to the nature and complexity of the problem, as well as the data collection circumstances. For this study, it has not been investigated how many samples we need to train a neural network, due to data availability and resources (neither for regular ML nor DL). However, we argue that more than one representation of each scenario is

Table 1

Explanation and comparisons of desired operational properties of a rigid automated safety compliance analysis system.

	Explanation	Machine learning approach	Rule-based approach
Transparency	Transparency means clearly stating what the system looks for (i.e., observations) that are required for a chunk of data to belong to a certain category.	Machine learning is identifying features that distinguish different classes in the training data automatically. It may identify features that are or are not obvious, but the user would not be able to know.	The nature of the rule-based approach is based on <i>if x then y</i> rules, which allow the user to follow the reason behind a verdict. The x can describe the observations that are needed for a predicate y to be true.
Explainability	It should be possible to make the system explain what observations led to the specific classification and vice versa to explain why it did not pick another class. This functionality allows the system user to prepare for the repair, enabling them to bring the necessary parts right away.	As it is not possible to gain access to the specific observations that describe a class, it is also not possible to gain access to the observations that lead to a specific prediction. However, explainability is a heavily investigated topic in ML research.	While a prediction is proved by the engine, its violation predicates can become true or false. This means that classification would necessarily include a set of observations that are identified, and these can be queried by the human operator to get an explanation of the verdict. The new observation can be added to the rule base. Additionally, it can be determined if the addition of the observation brings any ambiguity to the definitions of the other classes.
Extensibility	Extending the system to identify additional observations should not require a lot of effort. At the same time, it should be easy to identify if these introduce misclassifications or bias and the performance of old cases is decreases.	It is hard to know if an observation is considered by the machine learning network or not. Observations on a semantic level are not used in this approach and thus have not to be manually added. There are options for adding semantics in machine learning approaches, and a lower representation of the compliant scenarios would also make it less probable for the machine learning network to go for these classes, even though that is not a viable solution as it may affect classification negatively.	While building the rule-base and queries, it is possible to force the engine to consult all the described violations and, therefore let it try investigating if they exist. This means that the compliant classification is the last resort for the engine.
Pessimism	A pessimistic approach is a positive property for safety analysis. It is better to classify a compliant scenario as a faulty non-compliant rather than non-compliant as compliant. It is comparable to assuming it is a faulty scenario and trying to prove it differently by finding the necessary observations.		

needed for training, considering the training sample size described in the related work. While we get a high level of insight into what happens when the computer makes the decisions in the rule-based approach, we are also trading off some explorativeness. It is plausible that certain underlying information in the data is not explicitly known by the human experts whose domain knowledge is being captured and formalized, which the ML approach could potentially exploit. The issue is that we do not know, and we do not have full control. This means that if a data-driven system behaves differently than expected, we can only collect more data and continue training as users [55,56]. Therefore, this study uses a rule-based white-box approach [57], where all the decisions can be explained and supported by concrete evidence.

Based on the decision to go for a rule-based approach, the subsequent classification process relies on the formalized safety regulation queries, which, as the name hints, are created based on the regulation. Queries (questions) such as “Does a railing have three boards?”, “Are the three boards parallel?” and “Are the distance between all boards less than the maximum allowable distance?” can be formalized and answered by the query engine. This study specifically relies on Prolog [58,59], to answer these queries. The formalized safety query and classification process are further described in the following Section 4. This covers the facts (i.e., observations) and the predicates (i.e., formalized domain rules) that are captured in the knowledge base to allow for classification and compliance reporting.

4. Rules and domain formalization procedure

The following describes the principal steps of preparing our rule-based guardrail compliance checking system, to determine if a semantic, knowledge-driven approach can be used to automate and improve safety inspection frequency. The first step is to predefine the scenarios representing the conceptual conditions where guardrails can be found on the construction site. The second step is to formalize the concept from the predefined scenarios to define their violations and the observations that are expected for those to exist. The third step is to get BIM model data and PCD to a logic fact representation in a way that is as simple and less biased as possible. The objective is to employ a semantic, knowledge-based approach when assessing a guardrail. Thus, we need to enable the computer to separate the toe, middle, and top boards from each other. This task is trivial for a human, but for a computer, it is necessary to describe the logic behind it. Listing 1 shows how the top board is identified from the list of *objects*. The PCD formalization procedure generates the object facts and consists of a way to identify the *object-type* (i.e., either board or pole), shown in line 2. Line 3 captures a way to identify the relationship between the *object_id*, in this case, *Board_id*, and the *File_id* (i.e., the individually cut PCD). In line 4, the *Y-coordinates* are extracted from the line segment called *seg*, which is connected to the *Board_id*. The regulation often mentions a minimum height (*Required_height*) and a maximum distance between boards and poles. In line 6, the upper third of the railing is identified and stored in the *Start* and *End* variable, which is then used in our *is_between* predicate that takes care of the comparison (i.e., if Y1 OR Y2 is in the upper third of the railing). The disjunction is because we still associate a board as the top board if at least one of the ends is at the top, which is handy if a top board falls down behind another. This enables us to say that the top board is sloped because the board is identified as the top board. If the top board predicate is satisfiable, we have an association between a *File_id* and the *Board_id*.

```
1. top_board(File_id, Board_id):-
2.   object(type, Board_id, board),
3.   object(file_id, Board_id, File_id),
4.   object(geometry, Board_id, seg(_, Y1, _, Y2)),
5.   railing_height(File_id, Required_height),
6.   Start is Required_height*2/3, End is Required_height*3/3,
7.   (is_between(Start, Y1, End);is_between(Start, Y2, End)).
```

Listing 1. Predicate for identification of the top board (prolog code)

A similar approach has been used to identify the middle- and toe boards. The main difference is the *Start* and *End* variables being set to [1/3–2/3] and [0–1/3] for the middle and toe board identification, respectively. The thirds represent the inaccuracy in temporary equipment but are easily replaced by concrete numbers from the regulation or based on domain expert experience. Additionally, the approach has been used for poles, where the identification is based on the X-coordinates and railing length instead of the Y-coordinates and height. It should be mentioned that it only makes sense to talk about a middle pole in the case, where the installation requires 3 poles and that the same goes for boards, thus it has been handled with an additional requirement.

Listing 2 shows an example of the missing board violations. It has been decided to index the boards from the top and down, which means that the violation should only be satisfiable when there is a demand for three or more boards.

```
1. violation(toe_board_missing, File_id):-
2.   file_record(_,File_id),
3.   min_num_boards(File_id, Req_B)
4.   Req_B>=3,
5.   not(toe_board(File_id,_)).
```

Listing 2. Predicate for toe board missing violation (prolog code)

Example: In the case of a window, where the height of the window removes the requirement of a toe and middle board, it is not correct to conclude that there is a violation in lacking these. Therefore, this information is extracted from the BIM model and associated with the *File_id*, shown in lines 3 and 4. Line 2 ensures that the predicate is tested for all the files, i.e., if there is a file record, it should be tested if the violation is satisfied. Line 5 checks if the toe board predicate is *not* satisfied (i.e., not satisfied), which would result in a violation for the specific *File_id*, i.e., the violation-predicate is true.

The violations that regard the length of the boards are implemented in both right- and left-oriented predicates to distinguish the **Part 1** and **Part 2** scenarios in Fig. 2. Listing 3 shows an example of implementing a right-oriented *board_too_short* predicate for the toe board. Line 2, similarly to the previous example, makes sure that the violation is checked if there is a file record. Lines 3 to 6 are making sure that there is a toe board associated to the *File_id*, and that the length of that board is *not* similar to the *Required_length*. The *similar*-predicate is satisfied if the given numbers are within the margin (in this case, 20 cm), and the negation would mean that the board is shorter than expected. Lines 7 and 8 are introduced to check if it is to the right or the left (for the left-oriented version, the check is $X1 > (Required_length - X2)$). The 20 cm is chosen to represent a gap that would not expose humans to fall and based on experience exist in real construction sites. The number can however easily be replaced by concrete numbers from the safety regulation or company practices.

```
1. violation(toe_board_too_short_right, File_id):-
2.   file_record(_,File_id),
3.   toe_board(File_id, Board_id),
4.   board_length(Board_id, Length),
5.   railing_length(File_id,Required_length),
6.   not(similar(Required_length,Length,20)),
7.   object(geometry, Board_id, seg(X1,_,X2,_)),
8.   X1<(Required_length -X2).
```

Listing 3. Predicate for toe board too short right violation (prolog code)

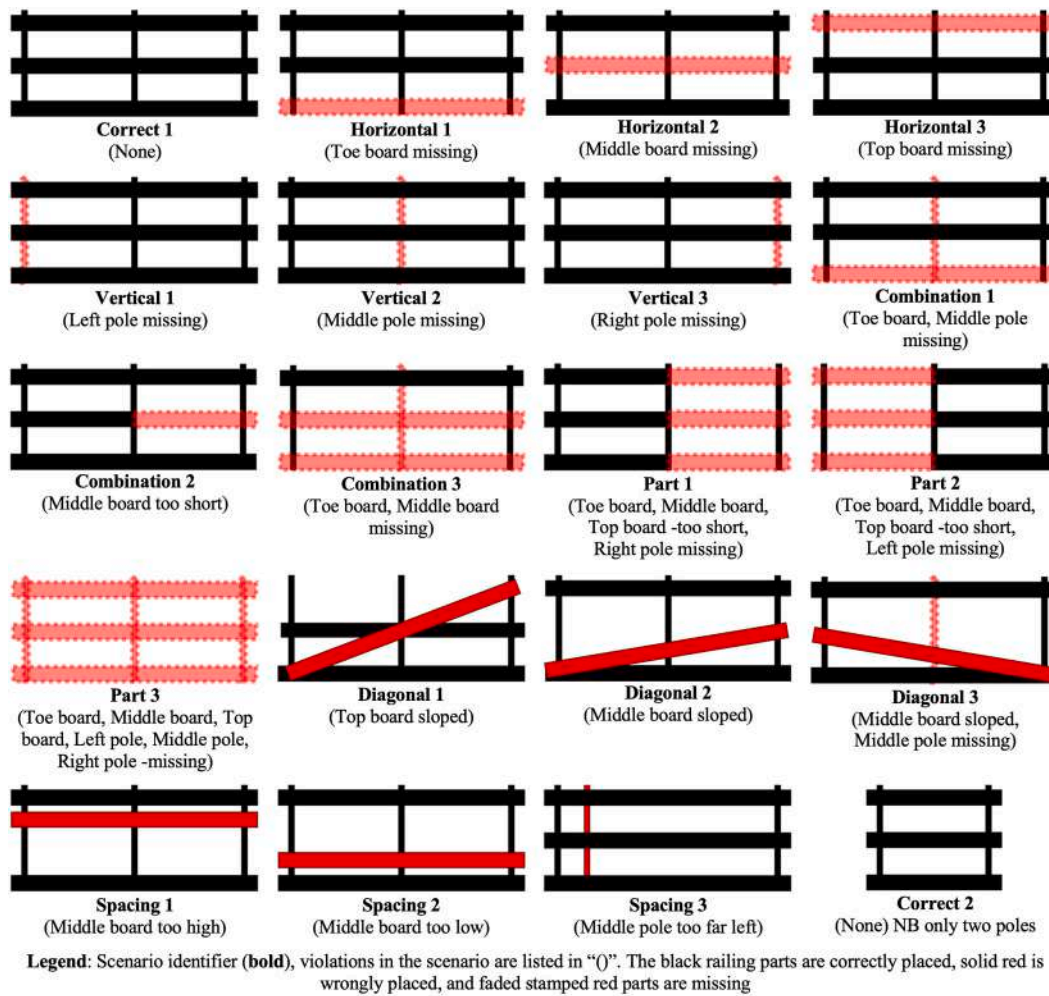


Fig. 2. Overview of the predefined scenarios used for rule-base creation. Those scenarios are later used as and referred to as classes for the guardrail classification.

Listing 4 shows the implementation of the violation, which checks if the middle board is installed too low (i.e., if the spacing between the top and middle is too big). In line 3 and 4, it is checked whether a middle board is expected, then Line 5 checks if there was *not* found any middle boards associated with the *File_id*. Line 6 then creates a set of the associated toe boards where the last parameter (`[_ , _ | _]`) is a way to check whether the set has two elements. Conceptually, we check if we have two toe boards and no middle boards. The middle board too high violation is implemented similarly, but with respect to the top board, i.e., expecting two top boards.

```

1. violation(middle_board_too_low, File_id):-
2.   file_record(_,File_id),
3.   min_num_boards(File_id, Req_B),
4.   Req_B>=2,
5.   not(middle_board(File_id,_)),
6.   setof(Object_id, toe_board(File_id, Object_id), [_ , _ | _]).

```

Listing 4. Predicate middle board too low (prolog code)

```

1. violation(top_board_sloped, File_id):-
2.   file_record(_,File_id),
3.   top_board(File_id,Board_id),
4.   board_direction(Board_id, TopVector),
5.   parallel_margin(Margin),
6.   (
7.     (
8.       toe_board(File_id, ToeBoard_id),
9.       board_direction(ToeBoard_id, ToeVec),
10.      not(similar_V(TopVector,ToeVec,Margin))
11.    );
12.    (
13.      not(toe_board(File_id,_)),
14.      not(similar_V(TopVector,(1.0,0.0),Margin))
15.    )
16.  ).

```

Listing 5. Predicate top board sloped (prolog code)

Listing 5 shows the implementation of the board sloped violation, specifically for the top board. There is a similar implementation for the middle board. Line 3 identifies the *Board_id* of the top board, and Line 4 extracts the direction vector of it. Line 5 extracts the *Margin*, which is set as a parameter in the instance creation and determines when vectors are considered parallel. Lines 6 to 16 hold an OR statement (*disjunction*), where the top board must be parallel to the toe board, which is assumed to be parallel to the surface on which the railing is placed. If no toe board

is associated with the railing instance, it must be assumed that the top board is parallel to the horizontal vector (i.e., $[1,0,0]$). The *similar_V* predicate is a predicate that has been made to compare vectors to the *Margin*. Conceptually it says that if a toe board exists and the top board's direction vector is not parallel to its direction vector, there is a violation, and if there is no toe board, the top board must be directly horizontal.

The above listings provide an overview of the predicates used to determine if the formalized railing scenarios include any violations and thus are incompliant with the regulation. The following section explains how the railing instances are formalized into the *object-facts*. Readers are as mentioned referred to the GitHub repository [60] for the remaining implementation of the rule-base.

4.1. Point cloud data collection and formalization

The previous section presented a selection of the rules in the rule base. In Listing 1, it was shown how the observations of objects were turned into observations of top, middle, and toe boards using logic inference. A similar approach is, as mentioned, done for the poles. The logic inference and the transformation from information to knowledge are, on purpose, kept in the logic part of our framework. This decision relaxes the information extraction requirement, keeping it simpler, and separates concerns. In this section it is described how the PCD was collected and processed into the observation needed in the inference and classification step, i.e., getting from the PCD to *object(File_id, Instance_id, Type, seg(X1, Y1, X2, Y2))*, that can be split into the observations used in Listing 1 (Lines 2 to 4). The involved steps are visualized in Fig. 4.

Fig. 3 provides an overview of the construction site, where predefined and validation scenarios were collected for this study. The top and bottom images show the progress during the data collection campaign (28th until 30th of May 2024), and the three images between are, in orange, an example of a predefined scenario, and in dark blue, examples of two validation scenarios. Both kinds of collection areas have been pointed out with the colored areas in the two overview images (top and bottom).

After the point cloud has been cropped, it is paired with the corresponding planned element from the SafeBIM, and information about the expected number of poles and boards is extracted, their geometry, and location. To continue the process, a rasterization can be made to remove the depth dimension, turning it into a two-dimensional (2D) representation of the 1 by 1 cm grid. In the rasterization process, it is checked whether there is a point in the grid for any slice of the depth, and if so, the coordinate is activated (set to one) or otherwise deactivated (set to zero), which helps to remove potential depth noise in the 2D-representation.

The skeletonization procedure [64] is applied to obtain a one-pixel-wide representation of the active areas in the rasterization before applying the Hough transform. The procedure erodes the input, which can be several pixels wide, and makes the individual components more distinguishable and easier to identify in the subsequent Hough transformation.

The Hough transformation [54] is carried out on the skeletonized rasterization in two steps. In the first step, the Hough transform is focused on the boards, which means that it is tasked to search for lines from 45° to 135° , as shown in the upper part of Fig. 5. Secondly, the same Hough transformation procedure is performed but focused on poles, which are expected to be found between -45° to 45° , with the convention that the y-axis is 0° , as shown in the lower part of Fig. 5. For both cases, rather distinct peaks are found due to the skeletonization procedure. As mentioned, it is known on forehand how many boards and poles the Hough transformation procedure should identify, as these are determined by the dimensions of the fall hazard and the regulation and modeled in the SafeBIM. However, these numbers can only be used as an upper boundary because it is unknown if the expected number of poles or boards is installed. In the example in Fig. 5, the Hough transform is tasked to look for three poles and three boards but can only find two of

each.

The output of the Hough transform is formed as regular line descriptors (i.e., a set of start coordinates and the slope of the lines). However, these line descriptors do not say anything about the continuation and length of the railing parts (i.e., there could be a situation where one meter of the top board is cut away but still identified as a predominant line in the Hough transformation).

```

Input: M, LDs (Rasterized railing and line descriptors)
Output: S (a set of start and end coordinates of consecutive parts)
1. for each Line_descriptor in LDs
2.   index= Line_descriptor[0]
3.   slope= Line_descriptor[1]
4.   start_coordinate = index
5.   while index.x <= M.shape.x and index.y <= M.shape.y
6.     search_vector := ones(5 * slope.x, 5 * slope.y)
7.     active_pixels := sum(M[index + search_vector])
8.     if active_pixels == 0:
9.       if index != start_coordinate:
10.        S := S^[[start_coordinate, index]]
11.        start_coordinate := index
12.        index := int(index+slope)
13.     if index != start_coordinate:
14.       S := S^[[start_coordinate, index]]
15.   return S

```

Listing 6. Binary pixel search algorithm (pseudo code)

Therefore, whether there are active pixels (pixels with value one) all along the identified lines must be checked. Listing 6 shows the algorithm for identifying the consecutive railing parts based on the rasterization (*M*), and line descriptors (*LDs*), which are the two inputs. The output, *S*, is a set of the start- and end-coordinates of the consecutive railing parts. Each *line_descriptor* in *LDs* consists of an *index* (the start coordinates of the line) and a *slope* (the normalized direction vector format of the line). Line 4 initiates the *start_coordinate* with the value of the *index*. The remainder is an iterative process, which continues while the search is still within the boundaries of the rasterized railing matrix. Line 6 creates the *search_vector* (an extended version of the slope that relaxes the requirements to the point cloud quality). Conceptually, it looks for active pixels within a five cm range and allows for small gaps. In line 7, the *search_vector* is applied at the current index, and the sum of ones is returned to the *active_pixel* variable. This process continues until the sum of pixels is zero, meaning there are no active points in the pixels at the *search_vector*, and the end of a part has been reached. The if-statement in line 9 ensure that the algorithm finds a part before it is allowed to reach the 'end'. Line 10 shows the concatenation (^) of the newly identified part's start and end-coordinate. Line 12 moves the *index* along the *slope* for the next iteration. Line 14 adds the part even if the end cannot be found within the boundaries of the rasterization, ensuring that parts are also considered if they extend the bounding box.

The last step of Fig. 5 is the formalization procedure, which turns the extracted information about the railing parts (i.e., boards and poles) into logic facts that can be used by the rule-base and the logic inference engine. Besides the railing parts, this step also formalized the basic SafeBIM information, such as the expected number of poles and boards and the expected height and length of the railing. With the rule base described previously and the recently added observations from the data, it is possible to build queries to investigate an instance from the construction site data. These queries are described in the following section and combine the regulation, best practices, and data into knowledge about the on-site installation of fall protection equipment.

4.2. Classification capabilities of proposed system

One of the strengths of the rule-based white-box approach is that it can provide a lot of insight into different categories of information for



Fig. 3. Data collection environment from showing two days of progress, predefined scenario collection area (orange), and the validation scenarios (V0–8) collection areas (dark blue). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

different use cases. Its flexibility allows the user to specify in the query what, and how, the results should be presented and sorted. This section describes some of the queries used during the development of the presented classification system and proposes other advancements that could be made to e other kinds of information.

In a situation where one would like to know where a specific violation is found based on a set of observations. To do so, we rely on the *violations(Violation, File_id)*, which is true for all the files that contain the specific violation. Depending on the instantiation of the *Violation*- and *File_id*-variables in the query, we can ask the engine to return all the violations for all the files, the violations for a specific file, all the files with this violation, or even if a specific file contains a specific violation.

The same approach has been used in the *file_violation(File_id, Set)*, shown in Listing 7, which identifies all the violations for a particular *File_id* and returns the *Set* of all associated violations. The *Set* is also used in later classification-related queries. The cut-operator (!) in Line 3 ensures that the engine does not backtrack if a solution has been found. Consequently, it does not return the empty set if it has already returned a set of violations. This is necessary as the **Correct 1** and **2** classes, as shown in Fig. 2, do not have any violations.

```
1. file_violations(File_id, Set):-
2.   file_record(_, File_id),
3.   setof(Vio, violation(Vio, File_id),Set),!; Set=[].
```

Listing 7. Predicate for listing all the violations of a file in a Set (prolog code)

Similarly, a predicate has been created that extracts the *Set* of the violations listed for a class in the ground truth for the predefined scenarios. This predicate is called *class_violations(Class_id, Set)*, where the *Class_id* refers to the bold text in Fig. 2, and the *Set* to the list of violations listed in “()”. The two predicates can be used to compare True Positive (TP), False Negative (FN), and False Positives (FP) for the individual instances, i.e., comparing what has been found (i.e., *file_violations*-predicate) and what should have been found (i.e., *class_violation*-predicate). The data and software implementation of the mentioned and additional predicates are freely shared on GitHub [60] for others to use. The additional predicates cover functionality to compute a fit based on the TP, FP, FN, and sort for the best fitting class, thereby, the file’s classification. The example that has been brought forward assembles sorting, isolation of specific details, and general overviews with sets.

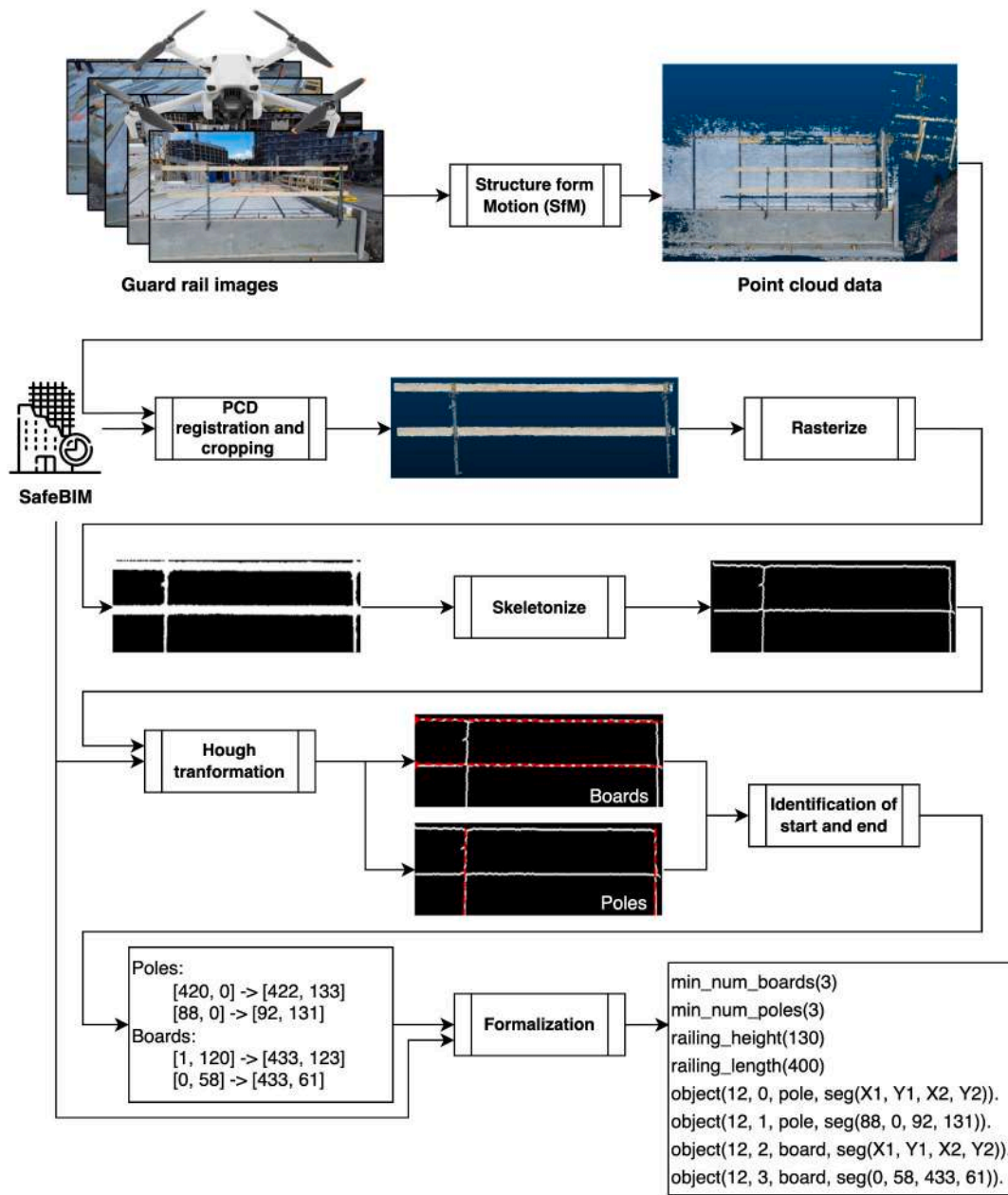


Fig. 4. Processing steps to get from point cloud data to observations in the knowledge base.

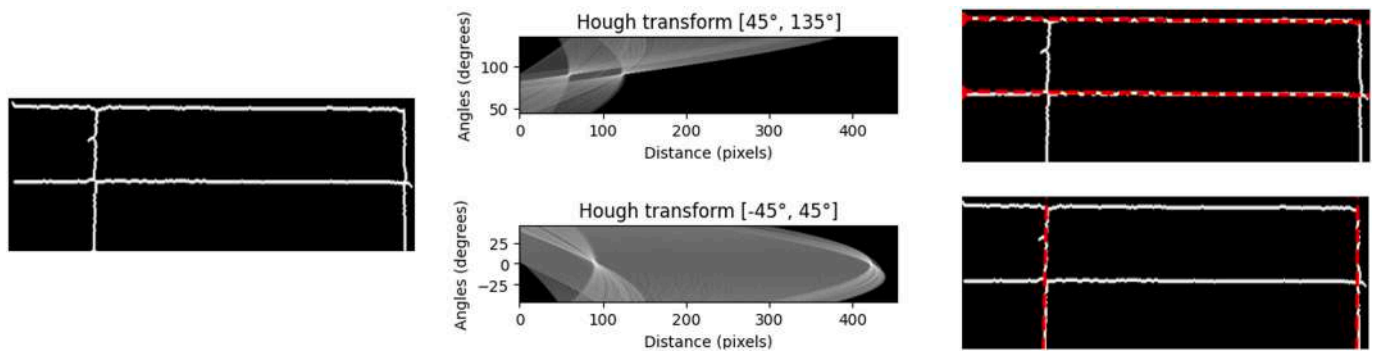


Fig. 5. Visualization of the Hough transformation step in Fig. 4.



Legend: identifier (**bold**), classification (*italic*), actual class (underlined), identified violations (true positives shown in ()), falsely identified violations (false positives shown in []), and the missing identified violations (False negatives shown in {})

Fig. 6. Visualization of the analysis results of the predefined scenarios. The scenarios are presented in the same sequence as in Fig. 2.

Suppose a company had records about the severity of the different violations, it would be straightforward to provide the information and sort the railings based on this information, which would allow the safety manager to address the most severe cases first before continuing to the others. Also, it would be possible to isolate the cases from a specific construction site area if that area would have increased activity during the previous and forthcoming days, or sorting the cases such that a more optimal route around the construction site can be selected while correcting the violations.

5. Results

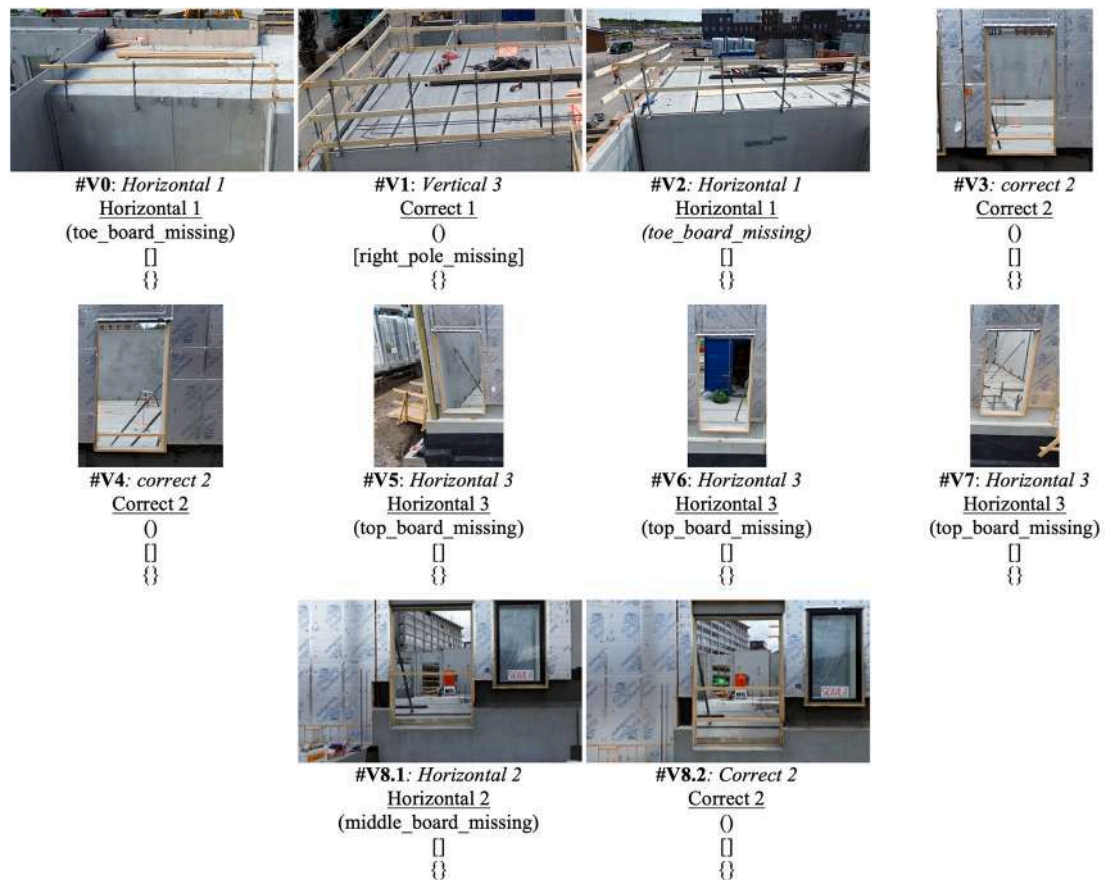
This section presents the obtained results of the study. The section is introduced with the results from the predefined scenarios presented in Fig. 2, which has been making the basis for what could be referred to as an adjusting phase. The adjustments have, for example, involved handling cases with less than three poles, in which case a middle pole does not make sense. Subsequently, the results of the rule-base have been validated on guardrailing installations found around the same construction site. The validation phase does, therefore, not include any staged scenarios but solely real installations of fall protection equipment set up at the construction site to protect the workers, as shown in Fig. 3.

5.1. Adjustment phase on predefined scenarios

This section presents the results of the proposed framework applied

to the predefined scenarios in the adjustment phase. It has been done in an iterative process, where flaws were identified and fixed, like a training procedure of other approaches, e.g., machine learning. The fixings mainly regarded adjustments of margins and incorrect formulation of the rules, such as falsely selected predicates. Thus, the adjustments were minor. To quickly assess the correctness of the developed rule base, a query has been created to classify all the files and afterward only show the files that are falsely classified compared to the ground truth. This query only works on the predefined scenarios and can be run when changes are performed in the rule base, e.g., when the system needs extension in case of new scenarios.

Fig. 6 shows an overview of the predefined scenarios. Each scenario is represented by one of the images used in SfM-procedure. The five lines below the images explain the identifier in bold (e.g., #P0) and the framework's classification in *italic* (e.g., *Correct 1*). The second line describes the ground truth classification underlined (e.g., Correct 1). The third line shown in “()” is the set of correctly identified violations, also called True Positives (TP). The fourth line shown in “[]” is the set of violations identified without existing, i.e., False Positives (FP). The fifth and last line shown in “{ }” is the set of violations that exist but were not identified, also referred to as False Negatives (FN). There remains an FP violation in #P4, which has not been removed through adjustment in the rule base because the top board is too short; thus, the “Diagonal 1” class was just not represented perfectly in reality as the board slide a bit to the left leaving it a too short in the right side. This means that compared to “Diagonal 1”, the best-fitted class, it has a false positive, but



Legend: identifier (**bold**), classification (*italic*), actual class (underlined), identified violations (true positives shown in ()), falsely identified violations (false positives shown in []), and the missing identified violations (False negatives shown in { })

Fig. 7. Visualization of the analysis results of the validation scenarios. Collected as shown in Fig. 3.

the violation exists in reality. The classification, therefore, remains true. This information that enables the concrete identification of the issue is fostered by the knowledge-driven approach, where classification and result can be reasoned through the Explainability ability mentioned in the methodology.

5.2. Validation phase on real scenarios

Like the predefined scenarios, Fig. 7 shows the results of the framework validation. Validation was performed on scenarios that were not staged but found on the construction site: the scenarios cover fall mitigation, different elevations of leading edges, and door and window holes.

In the figure, #V3 and #V4 are two windows in the same façade, likewise are #V5, #V6, and #V7. Validation scenarios #V8.1 and #V8.2 are recorded during the installation, where the middle board has not yet been installed in #V8.1, but in #V8.2, the guardrail becomes compliant after the installation. The figure shows that #V1 is misclassified as being a

scenario of *Vertical 3*, while being a compliant installation (i.e., Correct 1). #V1 is discussed further in the following Section 6.

5.3. Performance and feasibility of framework

To assess the performance of the proposed framework, it has been decided to use the common practice of assessing other detection strategies. The assessment is captured in Table 2 and shows the precision, recall, and combined F1-score for both individual violations and classes (i.e., the combination of violations). (See Table 3.)

To investigate the feasibility of the proposed method in a realistic construction scenario, it is necessary to increase the size of the problem. The data collection efforts are not considered in this feasibility study, as that step is not part of the concrete scope and should be investigated in the relevant studies. Therefore, this feasibility study considers only the classification process. This part of the study is, therefore, not about the correctness of the classification but the number of classifications. Thus, it was chosen to create replications of the predefined scenarios, which exploit the whole rule base and its predicates, i.e., all the considered violations. This should make the basis for a realistic extrapolation of the performance of the proposed rule-based compliance checking system.

Fig. 8 shows the results of the feasibility study, where the lower x-axis shows the number of replications, i.e., the how many times the 21 scenarios were replicated, the upper x-axis shows the number of guardrail instances, i.e., ($N = R \cdot 21$). On the y-axis, it is possible to read the processing times in seconds (left) and minutes (right) for each iteration of the feasible study. The number of replicas selected are shown in Table 3, resulting in the corresponding number of guardrailing instances and processing times. The study was performed on an Apple MacBook Pro M1 Max.

Table 2

Quantitative performance of the proposed classification framework.

	Predefined		Validation	
	violations	classes	violation	classes
Total	39	20	6	10
TP	38	20	5	9
FN	0	0	0	0
FP	1	0	1	1
Precision	97 %	100 %	83 %	90 %
Recall	100 %	100 %	100 %	100 %
F1 score	99 %	100 %	91 %	95 %

Table 3

Results of the feasibility study showing the number of replications, instances, and processing time.

Replica (R)	1	10	20	30	40	50	60	70	80	90	100	150	200	250	300
Instances (N)	21	210	420	630	840	1050	1260	1470	1680	1890	2100	3150	4200	5250	6300
Proc time (sec)	1	7	14	22	29	37	43	49	56	62	69	104	138	173	210
Replicas (R)	350	400	450	500	550	600	650	700	750	800	850	900	950	1000	10,000
Instances (N)	7350	8400	9450	1500	11,550	12,600	13,650	14,700	15,750	16,800	17,850	18,900	19,950	21,000	210,000
Proc time (sec)	242	276	314	361	399	423	452	485	546	583	619	654	688	718	7138

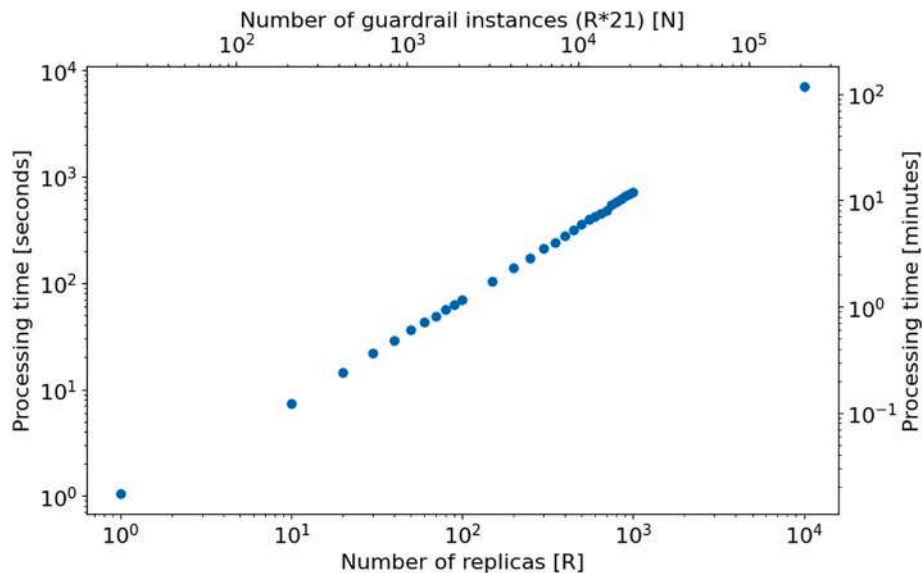


Fig. 8. Visualization of the feasibility study results showing a linear relationship between the processing time, and the number of instances (N) and replicas(R).

6. Discussion

The results show that it has been possible to represent all (i.e., 100 % F1-score in predefined classes) the cases in the predefined scenarios (Fig. 6). This has been accomplished by describing the combination of violations in each scenario and the logic behind them. While it is possible to identify all violations and perform a correct classification for all presented scenarios, it does not fully express the applicability of the proposed rule-based solution. Therefore, it was decided also to collect the images of existing scenarios around the construction site.

The validation results also yield promising capabilities for the framework, even though there is a misclassification, as shown in Fig. 7 (#V1). The overall F1-score for the validation classes is 95 %, as shown in Table 2. The cause of the misclassification was identified in the point cloud and later rasterization, where the pole does not appear very well. This can be reasoned by the fact that the scenario was collected only with eight images from almost the same drone elevation, which was insufficient to find image features on the pole in the SfM-procedure. Fig. 9(a) shows how poorly the rightmost pole is represented in rasterization; thus, it is fair that it is not identified by the following Hough transform. However, the fact that the remaining parts of Fig. 9(a) are identified is impressive. Fig. 9(b) and (c) are two other examples that show the robustness of the proposed rasterization, skeletonization, Hough transformation, and binary pixel search procedure, where discontinued and sparsely represented parts are still identified.

The proposed framework allowed us to use a knowledge-driven approach where observations of parts and their placement can be turned into eventual violations and classification. Even though a single violation is enough to make a guardrail non-compliant with the regulation, it was decided to classify them further into 21 classes, defined by the predefined scenarios presented in Fig. 2. These classes can help the safety manager query the rule and knowledge base in different levels of abstraction (i.e., observation, violation, and classification) in the same way as a manual assessment. The 21 classes that have been identified may not be exhaustive; however, they cover a wide selection of cases, which could be expected for a railing. Also, it is important to mention that the rule base has not been tightly coupled to the count of poles and boards of three but is dynamically extracted from the BIM model.

While it is a clear strength that, based on only 21 scenarios, the proposed system can start classifying actual guardrailings from a construction site, it is still not fully validated for the construction site as the validation should consist of several hundreds of scenarios that are checked by humans. Therefore, applying the proposed system cautiously and collecting more validation data is necessary. Still, the system has been made pessimistic, where a lack of information will turn out as a violation rather than optimistically believing that the unavailability of information is good. Additionally, the knowledge modeling of the rule base instead of a data-based approach, for example, machine learning, lowers the number of required input examples. The faulty and non-compliant scenarios are hard to collect in a live construction situation as these represent actual dangers to the workers. Therefore, collecting a

sufficiently sized and representing data set of hazardous situations is impractical in the construction industry. Therefore, it is strongly recommended that an automated compliance checking system like the one proposed is used as a second opinion tool until an extensive amount of classification has been performed successfully and trust has been built.

In terms of feasibility and scalability, this study provided results that show a linear tendency until at least 210,000 safety equipment instances, where 18 different violation types are considered. It was shown that the processing time for 2100 instances took around 69 s, which is sufficient considering the time it takes to collect the data, which by any means would be longer. The reason for including scenarios with what seems like unrealistic amounts of instances (i.e., 2100 to 210,000) was that the instances are not limited to guardrail instances but could also consider other types of equipment and even other types of safety hazards that could be checked against an extended version of the rule base. In this case, it would be necessary to carefully model which equipment types the individual rules apply to keep similar performance (i.e., to limit the search space). However, as the automated data collection effort was not considered in the scope of this work it should be noted that the feasibility also does not consider this part.

7. Conclusion and outlook

This paper presented a knowledge-driven strategy for automating compliance checking of construction safety guardrails. The approach considers reliability regarding defined parameters, scalability regarding realistic problem sizes, and feasibility regarding data requirements during data collection, analysis, and reporting. The guardrails are frequently used in construction, and their purpose is to prevent one of the deadliest hazards. It has been possible to achieve impressive performance in classifying predefined and validation scenarios, even with relatively low amounts of data available. Even though it was not possible to find all possible scenarios on the construction site in the validation phase, the ones found were identified and assessed correctly by the system when the data quality was sufficient. The semantic knowledge-based approach of compliance checking of guardrails has yielded promising results that could be applied in other cases of safety compliance checking. One of the strengths of the chosen approach is that it does not require many examples of each situation before being able to classify it. This is important as hazardous examples of guardrails are not straightforward to collect in a live construction situation as they represent actual hazards to the workers. Furthermore, our feasibility study shows that the rule-based approach performs the classification quickly and scales linearly until at least 210,000 guardrail instances. This also means the task of classification will not be the bottleneck of the proposed automated inspection and compliance checking framework.

The system correctly identifies compliance issues in all 21 predefined scenarios and the 9 validation scenarios (when data quality is adequate), which is sufficient to prove the initial applicability of the proposed approach. However, we will continue data collection of additional scenarios at construction sites in the future. The additional data collection and further validation may yield the need for additional classes and

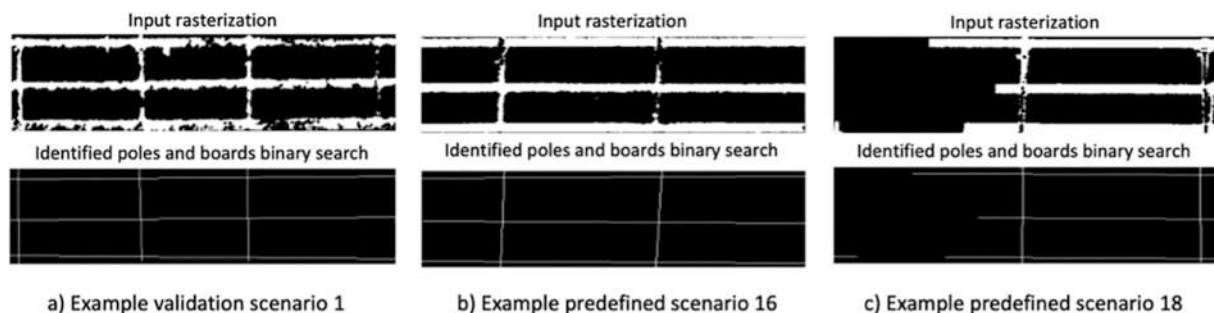


Fig. 9. Examples of identification performance in a) validation scenario #V1, b) predefined scenario #P16, and c) predefined scenario #P18.

other changes. Additionally, it would be interesting to investigate how cover panels and safety nets could be included in the solution to have a more complete solution. Moreover, it would be interesting to perform a study where the full pipeline is performed in sequence, potentially revealing new relevant topics to investigate and share with end users, as outlined in the Digital Twin for Construction Safety (DTCS) [65,66]. Our preprocessing code, knowledge base, and rule base have been freely shared for others to get inspired, confirm the results, investigate the remaining implementation, and even continue the work if desired. We encourage the community to investigate this further.

CRedit authorship contribution statement

K.W. Johansen: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization. **J. Teizer:** Writing – review & editing, Visualization, Supervision, Software, Resources, Project administration, Methodology, Funding acquisition, Formal analysis, Conceptualization. **C. Schultz:** Writing – review & editing.

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.

Acknowledgment

The research presented in this paper was funded by the European Union Horizon 2020 research and innovation program under grant agreement no. 958398 (BIM2TWIN) and no. 101058548 (BEEYOND-ERS). The authors thank Fira Oy and Sitedrive Oy for access to their construction site in Helsinki, Finland.

Data availability

Data is freely available on GitHub.

References

- [1] A. Pinto, I.L. Nunes, R.A. Ribeiro, Occupational risk assessment in construction industry – overview and reflection, *Saf. Sci.* 49 (2011) 616–624, <https://doi.org/10.1016/j.ssci.2011.01.003>.
- [2] BLS, National census of fatal occupational injuries in 2022, 2023, <https://www.bls.gov/news.release/pdf/cofi.pdf>.
- [3] J. Xu, C. Cheung, P. Manu, O. Ejowwomu, J. Too, Implementing safety leading indicators in construction: toward a proactive approach to safety management, *Saf. Sci.* 157 (2023) 105929, <https://doi.org/10.1016/j.ssci.2022.105929>.
- [4] E.L. Jacobsen, J. Teizer, Deep learning in construction: review of applications and potential avenues, *J. Comput. Civ. Eng.* 36 (2022) 03121001, [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0001010](https://doi.org/10.1061/(ASCE)CP.1943-5487.0001010).
- [5] W.F. Clocksin, C.S. Mellish, *Programming in Prolog*, Springer, Berlin, Heidelberg, 2003, <https://doi.org/10.1007/978-3-642-55481-0>.
- [6] V. Lifschitz, *Answer Set Programming*, Springer International Publishing, Cham, 2019, <https://doi.org/10.1007/978-3-030-24658-7>.
- [7] L. Brownston, R. Farrell, E. Kant, N. Martin, *Programming Expert Systems in OPS5: An Introduction to Rule-Based Programming*, Addison-Wesley Longman Publishing Co., Inc, USA, 1985, <https://doi.org/10.5555/3927>. ISBN: 0201106477.
- [8] T. Fruhwirth, *Constraint Handling Rules*, Cambridge University Press, 2009, <https://doi.org/10.1017/CBO9780511609886>. ISBN: 9780511609886.
- [9] Q. Sun, Y. Turkan, A BIM-based simulation framework for fire safety management and investigation of the critical factors affecting human evacuation performance, *Adv. Eng. Inform.* 44 (2020) 101093, <https://doi.org/10.1016/j.aei.2020.101093>.
- [10] C. Eastman, J. Lee, Y. Jeong, J. Lee, Automatic rule-based checking of building designs, *Autom. Constr.* 18 (2009) 1011–1033, <https://doi.org/10.1016/j.autcon.2009.07.002>.
- [11] B. Li, J. Fitzgerald, C. Schultz, Modelling the impacts of crowds on occupants in the built environment—a static, rule-based approach to human perception and movement, *Adv. Eng. Inform.* 51 (2022) 101452, <https://doi.org/10.1016/j.aei.2021.101452>.
- [12] J. Qi, R.R.A. Issa, J. Hinze, S. Olbina, Integration of Safety in Design through the Use of Building Information Modeling, 2012, pp. 698–705, [https://doi.org/10.1061/41182\(416\)86](https://doi.org/10.1061/41182(416)86).
- [13] J. Qi, R.R.A. Issa, S. Olbina, J. Hinze, Use of building information modeling in design to prevent construction worker falls, *J. Comput. Civ. Eng.* 28 (2014), [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000365](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000365). A4014008.
- [14] B. Li, C. Schultz, J. Teizer, O. Golovina, J. Melzner, Towards a unifying domain model of construction safety, health and well-being: SafeConDM, *Adv. Eng. Inform.* 51 (2022) 101487, <https://doi.org/10.1016/j.aei.2021.101487>.
- [15] K.W. Johansen, C. Schultz, J. Teizer, BIM-based fall hazard ontology and benchmark model for comparison of automated prevention through design approaches in construction safety, in: 29th Int. Workshop Intell. Comput. Eng. EG-ICE, 2022, <https://doi.org/10.7146/aul.455.c231>.
- [16] R. Takim, M.H. Zulkifli, A.H. Nawawi, Integration of automated safety rule checking (ASRC) system for safety planning BIM-based projects in Malaysia, *Procedia Soc. Behav. Sci.* 222 (2016) 103–110, <https://doi.org/10.1016/j.sbspro.2016.05.195>.
- [17] K. Kim, Y. Cho, S. Zhang, Integrating work sequences and temporary structures into safety planning: automated scaffolding-related safety hazard identification and prevention in BIM, *Autom. Constr.* 70 (2016), <https://doi.org/10.1016/j.autcon.2016.06.012>.
- [18] C.K.I.C. Ibrahim, P. Manu, S. Belayutham, A.-M. Mahamadu, M.F. Antwi-Afari, Design for safety (DfS) practice in construction engineering and management research: a review of current trends and future directions, *J. Build. Eng.* 52 (2022), <https://doi.org/10.1016/j.jobbe.2022.104352>.
- [19] Z. Jin, J. Gambatese, D. Liu, V. Dharmapalan, Using 4D BIM to assess construction risks during the design phase, *Eng. Constr. Architect. Manag.* 26 (2019) 2637–2654, <https://doi.org/10.1108/ECAM-09-2018-0379>.
- [20] S. Zhang, J. Teizer, J.-K. Lee, C.M. Eastman, M. Venugopal, Building information modeling (BIM) and safety: automatic safety checking of construction models and schedules, *Autom. Constr.* 29 (2013) 183–195, <https://doi.org/10.1016/j.autcon.2012.05.006>.
- [21] G. Tekbas, G. Guven, BIM-based automated safety review for fall prevention, in: S. Ofluoglu, O.O. Ozener, U. Isikdogu (Eds.), *Adv. Build. Inf. Model*, Springer International Publishing, Cham, 2020, pp. 80–90, https://doi.org/10.1007/978-3-030-42852-5_7.
- [22] S. Zhang, K. Sulankivi, M. Kiviniemi, I. Romo, C.M. Eastman, J. Teizer, BIM-based fall hazard identification and prevention in construction safety planning, *Saf. Sci.* 72 (2015) 31–45, <https://doi.org/10.1016/j.ssci.2014.08.001>.
- [23] K.W. Johansen, C. Schultz, J. Teizer, Hazard ontology and 4D benchmark model for facilitation of automated construction safety requirement analysis, *Comput. Aid. Civ. Infrastruct. Eng.* (2023), <https://doi.org/10.1111/mice.12988>.
- [24] V. Reja Kumar, K. Varghese, Q.P. Ha, Computer vision-based construction progress monitoring, *Autom. Constr.* 138 (2022) 104245, <https://doi.org/10.1016/j.autcon.2022.104245>.
- [25] M.H. Ibrahimkhil, X. Shen, K. Barati, C.C. Wang, Dynamic progress monitoring of masonry construction through mobile SLAM mapping and as-built modeling, *Buildings* 13 (2023) 930, <https://doi.org/10.3390/buildings13040930>.
- [26] A.S. Greeshma, J.B. Edayadiyil, Automated progress monitoring of construction projects using machine learning and image processing approach, *Mater. Today Proc.* 65 (2022) 554–563, <https://doi.org/10.1016/j.matpr.2022.03.137>.
- [27] M. Mehranfar, A. Braun, A. Bormann, From dense point clouds to semantic digital models: end-to-end AI-based automation procedure for Manhattan-world structures, *Autom. Constr.* 162 (2024) 105392, <https://doi.org/10.1016/j.autcon.2024.105392>.
- [28] D. Hu, V.J.L. Gan, C. Yin, Robot-assisted mobile scanning for automated 3D reconstruction and point cloud semantic segmentation of building interiors, *Autom. Constr.* 152 (2023) 104949, <https://doi.org/10.1016/j.autcon.2023.104949>.
- [29] Z. Kolar, H. Chen, X. Luo, Transfer learning and deep convolutional neural networks for safety guardrail detection in 2D images, *Autom. Constr.* 89 (2018) 58–70, <https://doi.org/10.1016/j.autcon.2018.01.003>.
- [30] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, 2015, <https://doi.org/10.48550/arXiv.1409.1556>.
- [31] Y. Li, B. Esmaeili, M. Gheisari, J. Kosecka, A. Rashidi, Using Unmanned Aerial Systems (UAS) for Assessing and Monitoring Fall Hazard Prevention Systems in High-rise Building Projects, 2022, <https://doi.org/10.48550/arXiv.2209.13137>.
- [32] E. Chian, W. Fang, Y.M. Goh, J. Tian, Computer vision approaches for detecting missing barricades, *Autom. Constr.* 131 (2021) 103862, <https://doi.org/10.1016/j.autcon.2021.103862>.
- [33] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, 2015, <https://doi.org/10.48550/arXiv.1505.04597>.
- [34] M. Gheisari, A. Rashidi, B. Esmaeili, Using Unmanned Aerial Systems for Automated Fall Hazard Monitoring, 2018, pp. 62–72, <https://doi.org/10.1061/9780784481288.007>.
- [35] Q. Wang, Automatic checks from 3D point cloud data for safety regulation compliance for scaffold work platforms, *Autom. Constr.* 104 (2019) 38–51, <https://doi.org/10.1016/j.autcon.2019.04.008>.
- [36] J. Kim, D. Chung, Y. Kim, H. Kim, Semantic segmentation of 3D point cloud data acquired from robot dog for scaffold monitoring, in: Dubai, UAE, 2021, <https://doi.org/10.22260/ISARC2021/0106>.
- [37] K. Kim, J. Teizer, Automatic design and planning of scaffolding systems using building information modeling, *Adv. Eng. Inform.* 28 (2014) 66–80, <https://doi.org/10.1016/j.aei.2013.12.002>.
- [38] B. Yang, B. Zhang, Q. Zhang, Z. Wang, M. Dong, T. Fang, Automatic detection of falling hazard from surveillance videos based on computer vision and building information modeling, *Struct. Infrastruct. Eng.* 18 (2022) 1049–1063, <https://doi.org/10.1080/15732479.2022.2039217>.

- [39] R. Amor, J. Dimyadi, The promise of automated compliance checking, *Dev. Built Environ.* 5 (2021) 100039, <https://doi.org/10.1016/j.dibe.2020.100039>.
- [40] W. Solihin, C. Eastman, A knowledge representation approach in BIM rule requirement analysis using the conceptual graph, in: *ITcon* 21, 2016, pp. 370–401. Special Issue CIBW78 20215, <https://www.itcon.org/2016/24>.
- [41] M. Bhatt, J. Lee, C. Schultz, CLP(QS): A Declarative Spatial Reasoning Framework, 2011, pp. 210–230, https://doi.org/10.1007/978-3-642-23196-4_12.
- [42] A. Borrmann, E. Rank, Topological analysis of 3D building models using a spatial query language, *Adv. Eng. Inform.* 23 (2009) 370–385, <https://doi.org/10.1016/j.aei.2009.06.001>.
- [43] T. Cooke, H. Lingard, N. Blismas, A. Stranieri, ToolSheDTM: the development and evaluation of a decision support tool for health and safety in construction design, *Eng. Constr. Archit. Manag.* 15 (2008) 336–351, <https://doi.org/10.1108/09699980810886847>.
- [44] K.W. Johansen, R. Nielsen, C. Schultz, J. Teizer, Automated activity and progress analysis based on non-monotonic reasoning of construction operations, *Smart Sustain. Built Environ.* 10 (2021) 457–486, <https://doi.org/10.1108/SASBE-03-2021-0044>.
- [45] B. Li, R.O. Nielsen, K.W. Johansen, J. Teizer, P.G. Larsen, C. Schultz, Towards digital twins for knowledge-driven construction progress and predictive safety analysis on a construction site, in: T. Margaria, B. Steffen (Eds.), *Leveraging Appl. Form. Methods Verification Valid. Tools Trends*, Springer International Publishing, Cham, 2021, pp. 153–174, https://doi.org/10.1007/978-3-030-83723-5_11.
- [46] K.W. Johansen, R.P. de Figueiredo, O. Golovina, J. Teizer, Autonomous safety barrier inspection in construction: An approach using unmanned aerial vehicles and safe BIM, in: *ISARC Proc.*, IAARC, 2021, pp. 629–636, <https://doi.org/10.22260/ISARC2021/0085>.
- [47] M. Bueno, F. Bosché, H. González-Jorge, J. Martínez-Sánchez, P. Arias, 4-plane congruent sets for automatic registration of as-is 3D point clouds with 3D BIM models, *Autom. Constr.* 89 (2018) 120–134, <https://doi.org/10.1016/j.autcon.2018.01.014>.
- [48] M. Kiviniemi, K. Sulankivi, K. Kähkönen, T. Mäkelä, M.-L. Merivirta, BIM-based Safety Management and Communication for Building Construction, VTT, ISBN 978951387782-8, <https://publications.vtt.fi/pdf/tiedotteet/2011/T2597.pdf>, 2011.
- [49] J. Teizer, K.W. Johansen, C. Schultz, The Concept of Digital Twin for Construction Safety, 2022, pp. 1156–1165, <https://doi.org/10.1061/9780784483961.121>.
- [50] S. Zhang, F. Boukamp, J. Teizer, Ontology-based semantic modeling of construction safety knowledge: towards automated safety planning for job hazard analysis (JHA), *Autom. Constr.* 52 (2015) 29–41, <https://doi.org/10.1016/j.autcon.2015.02.005>.
- [51] M. Namian, M. Khalid, G. Wang, Y. Turkan, Revealing safety risks of unmanned aerial vehicles in construction, *Transp. Res. Rec.* 2675 (2021) 334–347, <https://doi.org/10.1177/03611981211017134>.
- [52] BG-Bau, Absturzsicherungen auf Baustellen Seitenschutz/Absperrungen | BG BAU - Berufsgenossenschaft der Bauwirtschaft, <https://www.bgbau.de>, 2021 (accessed February 11, 2022).
- [53] OSHA, Infographic: OSHA Fall Protection Guidelines & Standards | Graphic Products, <https://www.graphicproducts.com/infographics/osha-fall-protection-guidelines-and-standards/>, 2022 (accessed February 11, 2022).
- [54] R.O. Duda, P.E. Hart, Use of the Hough transformation to detect lines and curves in pictures, *Commun. ACM* 15 (1972) 11–15, <https://doi.org/10.1145/361237.361242>.
- [55] E.L. Jacobsen, J. Teizer, S. Wandahl, I. Brilakis, Probabilistic forecasting of metrics for construction labor productivity monitoring, *ITCon* 29 (2024) 58–83, <https://doi.org/10.36680/j.itcon.2024.004>.
- [56] K. Sokol, A. Hepburn, R. Santos-Rodriguez, P. Flach, bLIMEy: Surrogate Prediction Explanations Beyond LIME, 2019, <https://doi.org/10.48550/arXiv.1910.13016>.
- [57] Z.-J. Zhou, G.-Y. Hu, C.-H. Hu, C.-L. Wen, L.-L. Chang, A survey of belief rule-base expert system, *IEEE Trans. Syst. Man Cybern. Syst.* 51 (2021) 4944–4958, <https://doi.org/10.1109/TSMC.2019.2944893>.
- [58] A. Colmerauer, P. Roussel, The birth of Prolog, in: *Hist. Program. Lang.—II*, Association for Computing Machinery, New York, NY, USA, 1996, pp. 331–367, <https://doi.org/10.1145/234286.1057820> (accessed September 4, 2023).
- [59] C. Freksa, Qualitative spatial reasoning, in: D.M. Mark, A.U. Frank (Eds.), *Cogn. Linguist. Asp. Geogr. Space*, Springer Netherlands, Dordrecht, 1991, pp. 361–372, https://doi.org/10.1007/978-94-011-2606-9_20.
- [60] K.W. Johansen, C. Schultz, J. Teizer, Automated inspection and compliance checking, 2023, <https://github.com/kakke14/Automated-inspection-and-compliance-checking> (accessed October 4, 2023).
- [61] S. Ullman, S. Brenner, The interpretation of structure from motion, *Proc. R. Soc. Lond. B Biol. Sci.* 203 (1997) 405–426, <https://doi.org/10.1098/rspb.1979.0006>.
- [62] S. Jia, C. Liu, X. Guan, H. Wu, D. Zeng, J. Guo, Bidirectional interaction between BIM and construction processes using a multisource geospatial data enabled point cloud model, *Autom. Constr.* 134 (2022) 104096, <https://doi.org/10.1016/j.autcon.2021.104096>.
- [63] F. Xue, W. Lu, K. Chen, C.J. Webster, BIM reconstruction from 3D point clouds: a semantic registration approach based on multimodal optimization and architectural design knowledge, *Adv. Eng. Inform.* 42 (2019) 100965, <https://doi.org/10.1016/j.aei.2019.100965>.
- [64] W. Abu-Ain, S.N.H.S. Abdullah, B. Bataineh, T. Abu-Ain, K. Omar, Skeletonization algorithm for binary images, *Procedia Technol.* 11 (2013) 704–709, <https://doi.org/10.1016/j.protcy.2013.12.248>.
- [65] J. Teizer, K.W. Johansen, C.L. Schultz, K. Speiser, K. Hong, O. Golovina, A digital twin model for advancing construction safety, in: J. Fottner, K. Nübel, D. Matt (Eds.), *CLEAR 2023, Lecture Notes in Civil Engineering (LNCE)* vol. 390, Springer, 2024, pp. 201–212, https://doi.org/10.1007/978-3-031-44021-2_22. ISBN: 978-3-031-44020-5.
- [66] K.W. Johansen, J. Teizer, Digital Twin for Construction Safety Modules in the Plan-Do-Check-Act Cycle of Events and Information Exchange, 41st International Conference of CIB W78, Marrakech, Morocco, 2024. ISSN: 2706-6568, <http://itc.scix.net/paper/w78-2024-130>.