The European Master in Building Information Modelling is a joint initiative of:

Universidade do Minho

UNIVERZA V LJUBLJANI

Exploring agent-based models for construction planning using partial 4D BIM

Agnes Oliver Odrangwa Baka'di

BIM A+

UMinho | 2025

Universidade do Minho
Escola de Engenharia

Agnes Oliver Odrangwa Baka'di

Exploring agent-based models for construction planning using partial 4D BIM

BIM A+ European Master in Building Information Modelling

September 2025

Universidade do Minho
Escola de Engenharia

Agnes Oliver Odrangwa Baka'di

Exploring agent-based models for construction planning using partial 4D BIM models

BIM A+ European Master in Building Information Modelling

Master Dissertation
European Master in Building Information Modelling

Work conducted under supervision of:
Manuel Afonso Parente

September 2025

AUTHORSHIP RIGHTS AND CONDITIONS OF USE OF THE WORK BY THIRD PARTIES

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor and tutor, Professor Manuel Parente and Bruno Muniz, for their guidance, encouragement, and constructive feedback throughout this research. I am also grateful to my colleagues and friends in the BIM A+ programme, whose support and discussions have greatly enriched this journey.

Finally, I wish to thank my family for their unwavering love, patience, and encouragement, without which this work would not have been possible.

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

RESUMO

Título traduzido: Exploração de Modelos Baseados em Agentes para o Planeamento da Construção Usando Modelos Parciais 4D de BIM

A indústria da construção continua a enfrentar desafios relacionados com riscos de segurança espacial, em particular em contextos onde pavimentos, barreiras e transições estruturais permanecem incompletos. Embora o Building Information Modelling (BIM) ofereça uma base digital para a coordenação espacial, frequentemente carece de inteligência incorporada capaz de identificar autonomamente condições perigosas em obra. Esta dissertação explora a integração do reinforcement learning (RL) com modelos parciais de 4D BIM num ambiente de motor de jogo, de forma a simular um agente inteligente capaz de navegar, perceber e interagir autonomamente com elementos espaciais perigosos.

Foi desenvolvido um ambiente de simulação personalizado utilizando o Godot Engine, no qual foi importada geometria parcialmente construída derivada de modelos BIM. O ambiente apresenta condições de risco frequentemente associadas a acidentes por queda, tais como beirais abertos, vazios e caixas de escadas. Um agente único foi treinado através do Proximal Policy Optimization (PPO), com perceção baseada em sensores e uma função de recompensa concebida para promover a interação com transições inseguras. Ao contrário de estruturas convencionais baseadas na sobrevivência, este projeto inverteu a lógica: a queda foi recompensada e não penalizada, incentivando o agente a simular um comportamento de procura de perigos em vez de evitamento. A simulação teve como objetivo explorar a viabilidade de utilização de agentes RL como inspetores digitais de segurança capazes de identificar elementos inseguros sem recurso a etiquetagem semântica prévia. As expectativas principais incluíram a navegação entre múltiplos pisos, uma exploração espacial diversificada e a aprendizagem de políticas orientadas para riscos. Os resultados indicaram que, após o treino, o agente conseguiu aprender a atravessar geometria complexa e a interagir repetidamente com perigos não etiquetados utilizando apenas entradas de sensores de baixo nível e feedback ambiental.

Esta investigação oferece uma nova perspetiva sobre a aplicação do reinforcement learning na simulação de segurança em construção, reposicionando os agentes de executores de tarefas para exploradores diagnósticos. Destaca ainda o potencial da combinação entre BIM e sistemas de agentes inteligentes para apoiar o planeamento proativo da segurança em ambientes digitais de construção.

Palavras chave: Building Information Modelling (BIM), Deteção de Perigos, Motores de Jogo, Planeamento da Construção, Reinforcement Learning

ABSTRACT

The construction industry continues to grapple with spatial safety risks, particularly where floors, barrier and structural transitions remain incomplete. While Building Information Modelling (BIM) offers a digital foundation for spatial coordination, it often lacks embedded intelligence capable of autonomously identifying hazardous site conditions. This dissertation explores the integration of reinforcement learning (RL) with partial 4D BIM models in a game engine environment to simulate an intelligent agent capable of autonomously navigating, perceiving and interacting with hazardous spatial features.

A custom simulation environment was developed using the Godot Engine, into which partially constructed BIM-derived geometry was imported. The environment features hazardous conditions often associated with fall-related accidents such as open ledges, open voids and stairwells. A single agent was trained using Proximal Policy Optimization (PPO), with sensor-based perception and a reward function designed to promote interaction with unsafe transitions. Departing from conventional survival-based frameworks, this project inverted the logic: falling was rewarded and not penalised which encouraged the agent to simulate hazard-seeking rather than avoidance. The simulation aimed to explore the feasibility of using RL agents as digital safety inspectors capable of identifying unsafe features without prior semantic tagging. Key expectations included multi-floor navigation, spatially diverse exploration and hazard-focused policy learning. Results indicated that, after training, the agent could learn to to traverse complex geometry and repeatedly interact with unlabelled hazard using only low-level sensor input and environmental feedback.

This research offers a novel framing of reinforcement learning in construction safety simulation, repositioning agents from task-executors to diagnostic explorers. It highlights the potential of combining BIM and intelligent agent systems to support proactive safety planning in digital construction environments.

Keywords: Building Information Modelling (BIM), Game Engines, Hazard Detection, Reinforcement Learning, Safety Planning

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

2D	Two-Dimensional
3D	Three-Dimensional
4D	Four-Dimensional
ABM	Agent-Based Modelling
AEC	Architecture, Engineering and Construction
AI	Artificial Intelligence
API	Application Programming Interface
BEXEL[1]	BEXEL Manager (construction management software)
BIM	Building Information Modelling
CAD	Computer-Aided Design
CVPRW	Computer Vision and Pattern Recognition Workshops
ICML	International Conference on Machine Learning
IEEE	Institute of Electrical and Electronics Engineers
IFC	Industry Foundation Classes
ILO	International Labour Organization
MDP	Markov Decision Process
MIT	Massachusetts Institute of Technology
ML	Machine Learning
NLP	Natural Language Processing
OSHA	Occupational Safety and Health Administration
PPO	Proximal Policy Optimization
RGB	Red, Green, Blue (colour model)
RL	Reinforcement Learning
SMARLA[1]	Smart Reinforcement Learning in Architecture
VR	Virtual Reality

Notes:

[1] Not in the public domain; specific to software/conference/initiative.

1. INTRODUCTION

According to the International Labour Organization (ILO, 2023), more than 20% of occupational fatalities worldwide are construction-related. The construction industry remains one of the most hazardous sectors globally, as workers are frequently exposed to dynamic site conditions, heights and heavy machinery. Despite notable advancements in legislations and safety management systems, the early detection and mitigation of hazards during the project planning stages continue to be a persistent challenge (Hinze, 2006; Guo et al., 2020).

The ongoing digital transformation within the construction industry has positioned Building Information Modelling (BIM) as a cornerstone technology. This is due to its ability to provide a digital representation of built assets that is data-rich and in turn supporting the integration of scheduling, cost, and performance data throughout the asset lifecycle (Eastman et al., 2011; Bryde et al., 2013). Its application during design and planning stages has led to improvements in coordination, clash detection and construction logistics. Importantly, BIM has also opened new avenues for embedding safety considerations earlier in the project timeline, particularly through 4D BIM, which links model elements to time-based construction sequences (Wang, Chong and Zhang, 2016).

To fully appreciative the level of potential of BIM in regard to safety planning, it is prudent to first reflect on the practices that were previously undertaken. Traditionally, safety planning was predominantly based on reactive pre-planned practices and reliance on static checklists that were subject to human interpretation and experience (Zhou et al., 2012; Chi et al., 2014). As Hinze (2006) notes, traditional safety management lacked the spatial and temporal awareness needed for proactive hazard identification, especially in dynamic, multi-phase projects. BIM offers new opportunities to embed safety planning into the digital design process (Zhang et al., 2013; Kim et al., 2013). The visualisation of structural and temporal data allows for earlier identification of spatial conflicts and risk-prone activities. Safety-related elements such as scaffolding, guardrails, and restricted access zones can be visualised, validated, and updated in near-real-time.

However, many current practices are limited to static safety checks or compliance-driven rule applications. For example, in platforms such as Navisworks or Solibri, fall protection assessments may be semi-automated by predefined scripts that detect missing edge barriers or unsafe working platforms (Zhou et al., 2012; Dirgen Tözer et al., 2024). Consequently, despite their usefulness, these applications provide limited insight on how risks evolve over time or their response to environmental changes. Hazards such as temporary voids, partial staircases or incomplete access paths may go undetected until construction is already in progress (Amer et al., 2023).

Prior research has explored the application of BIM for automated identification of hazards. For instance, a system was developed for the detection of fall risks and unsafe zones by Zhang et al. (2013) and Kim and Chi (2019). While it proved effective in the visualization of hazards, the systems fell short in the ability to model adaptive worker behaviours simulation or emergent risks. Agent-Based Modelling (ABM), which offers decentralized simulations of worker-environment interactions (Bonabeau, 2002;

Lu & Olofsson, 2014), addresses some of the limitations but due to its assumption of deterministic behaviours, limits realism. Reinforcement Learning (RL), a subfield of artificial intelligence, address the limitations mentioned above by allowing agents to learn optimal behaviours through interaction with the environment and guided by a system of rewards and penalties (Sutton and Barto, 2018). In the construction context, Guan et al. (2021) and Lee et al. (2022) applied the navigation aspect in the BIM environments, but the implementation still remained isolated and rarely integrated fully with broader BIM workflows and evolving site conditions. Applications that train agents to explore environments in a bid to proactively seek, identify and interact with hazards remain scarce. Game engines such as Unity, Unreal Engine or Godot, enable for simulations that integrate RL with ABM and BIM presenting a potential for the development of intelligent and safety-aware simulations. These platforms provide real-time physics simulation, 3D rendering and programmable agent control. These factors make them ideal for the testing and visualization of agent behaviour in dynamic environments (Alves and Junior, 2020; Fang et al., 2020). Afsari, Eastman and Shelden (2021) highlight the unique value of game engines in the conversion of static BIM models into interactive digital environments that support behavioural learning and hazard visualization.

This dissertation proposes for the integration of BIM and RL within a game-engine environment to simulate proactive, behaviour-driven safety assessment in construction. Godot was chosen for this research and was used to host a virtual construction environment derived from a partial 4D BIM model enabling for the training of a reinforcement learning agent in the exploration, identification and logging of unsafe conditions as they emerged.

To date, a unified framework that combines Building Information Modelling (BIM), Reinforcement Learning (RL) and game engine-based simulation for proactive safety assessment in evolving construction environments remains unexplored in both academic and practical domains. This study aims to address the gap by assimilating these methods into a single simulation framework that moves beyond static assessments toward dynamic, behaviour-driven safety planning.

The proposed framework is not limited to navigational pathfinding but is designed to enable proactive hazard discovery. It seeks to simulate how intelligent agents that are trained via reinforcement learning, interact with incomplete and hazardous spatial conditions in construction environments derived from partial 4D BIM models. The ultimate intention is to identify how such agents can be used to detect high-risk areas, including unguarded edges, incomplete floor slabs and unprotected voids, thereby supporting stakeholders during early-stage safety reviews and design decision-making. Unlike conventional training approaches, where agents are programmed to avoid danger, the agent in this study is intentionally trained to seek out and engage with unsafe conditions. This offers a novel lens through which to simulate inspection behaviour and enhance planning foresight.

For the achievement of the research aim above, the objectives identified are as:

- To critically review the current state of BIM, RL and simulation technologies in relation to simulation of construction planning and safety so as to identify gaps in addressing safety scenarios.

- To develop a technical workflow for the translation of BIM-derived spatial data into a game engine-based environment that has the capability of supporting autonomous agent interaction and hazard simulation.
- To implement an autonomous agent capable of learning safety-aware navigation policies within a partially constructed 4D BIM environment, including the ability to identify and log hazardous conditions.
- To evaluate the performance of the RL agent in the risks identification, measuring hazard detection accuracy, adaptability, and comparative safety outcomes.
- To assess the framework's implications for proactive planning, hazard mitigation, and early design decision-making in real-world projects.

This study explores the integration of Building Information Modelling (BIM), reinforcement learning (RL) and game engine-based environments within a unified simulation framework in a bid to simulate behaviours related to safety in a 4D environment that is partially constructed. The simulation focuses on the early to mid-stage structural construction where significant fall hazards caused by temporary conditions such as open ledges, incomplete floors and stair voids, are most prominent.

The virtual environment will be based on simplified yet representative construction scenarios derived from imported BIM data and implemented in the Godot game engine. The reinforcement learning agent shall be trained to navigate these evolving spaces and identify unsafe features and while also using staircases for vertical movement. However, while stair structures are present, this work does not guarantee specific behaviours related to stairs (e.g., falling down a stairwell) unless explicitly demonstrated in the results section.

Significantly, this research is positioned as a proof of concept. Its main objective is to demonstrate the feasibility and usefulness in the transformation of BIM models into playable, learnable simulations for proactive safety assessments. Therefore, the focus is not on the refinement or optimization of the RL agent’s learning parameters. Instead, rather than focusing on the agent’s training efficiency, default RL configurations are engaged and the agent’s behaviour is evaluated based on its interaction with the environment.

Despite the proposed approach introducing a novel integration and modelling of behaviour, it is subject to several limitations. Due to the fact that the environment is a mere abstract, the simulation may not capture the complete spatial and operative complexity of real life active construction site. Limitations may stem from hardware and time, restricting the true scale of depth of RL agent training. Behavioural realism is limited as the validation of the agent behaviour occurs through simulated performance metrics and not through direct comparison with human site data. Additionally, the framework is primarily centered on fall-related hazards, excluding other risk categories such as equipment, collisions or handling of material. Finally, as with many efforts involving BIM, Interoperability remains a challenge due to the need for data translation between modelling platforms and simulation engines.

This dissertation is structured across five main chapters, each addressing a distinct stage of the research process while collectively advancing the goal of simulating hazard-seeking agents in partially constructed BIM-derived environments.

Chapter 1 establishes the context, rationale and scope of the study. It introduces the research problem, objectives and key contributions, situating the investigation within the domains of construction safety, digital modelling and reinforcement learning.

Chapter 2 provides a critical review of existing literature on Building Information Modelling (BIM), 4D simulation and reinforcement learning applications as well as game engines in safety-related environments. It identifies knowledge gaps and conceptual foundations that support the formulation of this research.

Chapter 3 details the procedural steps taken to implement the simulation environment. It covers BIM model preparation, Godot engine setup, agent control configuration and training routines. The methodology is structured to ensure reproducibility and to reflect a clear alignment between design intent and technical execution.

Chapter 4 presents the outcomes of the trained agent's behaviour, comparing actual simulation performance against the expectations set out in earlier chapters and existing practices. It evaluates the emergence of hazard-seeking behaviour, navigation patterns and simulation validity. This chapter also discusses observed limitations and offers recommendations for improving safety simulations.

Chapter 5 offers a reflection on the research journey, summarising the methodological contributions and core findings. It considers the implications of using reinforcement learning within digital construction models and outlines future opportunities for enhancing proactive safety planning using intelligent agents.

2. LITERATURE REVIEW

Building on the rationale established in the preceding chapter, this literature review critically examines the current body of research at the intersection of Building Information Modelling (BIM), Reinforcement Learning (RL), and game engine simulation in the context of construction safety. The aim is to explore how these technologies have been applied individually or in combination, to model, simulate, or enhance safety assessment in construction environments. Particular attention is paid to their methodological strengths and limitations, as well as to the key research gaps that persist in the simulation of adaptive safety behaviours.

While Agent-Based Modelling (ABM) has featured prominently in earlier safety simulation studies, especially those focused on modelling worker-site interactions, it is typically constrained by rule-based logic, limiting its capacity to simulate emergent or adaptive behaviours in dynamic environments (Bonabeau, 2002; Lu and Olofsson, 2014). In contrast, the emergence of Reinforcement Learning (RL) presents a promising alternative. By enabling agents to learn optimal behaviours through iterative interaction with their environment, RL avoids the need for manually programmed heuristics and opens pathways for more realistic behavioural simulations (Sutton and Barto, 2018).

This review is structured around three core thematic areas that underpin the development of a hazard-seeking simulation framework for construction planning:

- **Building Information Modelling (BIM):** The use of BIM for hazard representation, safety planning, and spatial data integration in construction environments.
- **Reinforcement Learning (RL):** The application of RL to enable agents to learn navigation strategies, respond to environmental hazards, and adapt to spatial changes.
- **Game Engines:** The role of simulation platforms such as Unity, Unreal Engine, and Godot in visualising BIM-derived environments and training intelligent agents.

The literature reviewed spans peer-reviewed journal articles, conference proceedings, and academic theses published between 2000 and 2025. Priority was given to research with demonstrated application to safety in construction, as well as studies that integrate simulation, behavioural modelling, or autonomous agent training. Targeted keyword searches included terms such as “BIM for safety”, “agent-based modelling in construction”, “reinforcement learning for hazard detection”, and “game engine simulations in construction safety”.

A summary of the reviewed literature, including their thematic relevance and contributions, is provided in **Table 1**.

Table 1 - Summary of reviewed Literature on BIM, ABM and RL

Author(s) & Year	Title	BIM	ABM	RL	Game Engine
Afsari et al. (2017)	Interoperability challenges in BIM-to-Game Engine pipelines	✓			✓
Afsari et al. (2021)	Utilising game engines for BIM-based simulation	✓			✓
Alves and Junior (2020)	Interactive simulations using Godot for architectural education				✓
Amer et al. (2023)	Automated construction hazard identification and prevention using NLP and BIM integration	✓			
Amodei et al. (2016)	Agent-based modeling: Methods and techniques for simulating human systems		✓		
Bonabeau (2002)	Agent-based modelling: Methods and techniques for simulating human systems		✓		
Chen et al. (2022)	Integrating reinforcement learning and ABM for adaptive safety simulations				✓
Chi et al. (2014)	Evaluating agent navigation through platform and stair constraints in virtual construction sites		✓		
Diniz et al. (2022)	Using open-source game engines in urban planning education				✓
Dirgen Tözer et al. (2024)	Safer designs with BIM-based fall hazard identification and accident prevention	✓			
Fang et al. (2020)	Comparative evaluation of Unity and Unreal Engine for construction safety visualisation				✓
Gao et al. (2021)	Reinforcement learning in interactive construction simulations using Unity			✓	✓

Author(s) & Year	Title	BIM	ABM	RL	Game Engine
Gao et al. (2021)	Integrating reinforcement learning and virtual reality for intelligent construction training environments				✓
Guan et al. (2021)	RL-based safety-aware pathfinding for construction robots in 3D BIM environments	✓		✓	✓
Guo et al. (2012)	A 4D model for tower crane safety planning	✓			
Guo and Yiu (2016)	Evacuation simulation in high-rise construction: An agent-based approach		✓		
Guo et al. (2020)	Predicting safety behaviour in the construction industry: Development and test of an integrative model			✓	
Hardin and McCool (2015)	BIM and Construction Management	✓			
Hsu et al. (2021)	Reach-avoid reinforcement learning with safety guarantees			✓	
Khalili and Helander (2020)	Crane path optimization using reinforcement learning			✓	
Kim and Chi (2019)	Hazard simulation using 4D BIM for proactive safety planning	✓			
Kim et al. (2013)	Automated information retrieval for hazard identification and safety compliance using BIM	✓			
Koo and Fischer (2000)	Feasibility study of 4D CAD in commercial construction	✓			
Konda and Tsitsiklis (2000)	Actor-critic algorithms		✓		
Leike et al. (2017)	AI Safety Gridworlds			✓	

Author(s) & Year	Title	BIM	ABM	RL	Game Engine
Liu et al. (2015)	Integration of BIM and agent-based modelling for construction site safety planning	✓	✓		
Liu et al. (2018)	Agent-based simulation of pedestrian–vehicle conflicts in construction zones		✓		
Lu and Olofsson (2014)	Building information modelling and planning: a 4D safety perspective	✓			
Lu and Olofsson (2014)	Building information modelling and agent-based modelling integration for construction safety analysis	✓	✓		
Lu et al. (2015)	Rule-based detection of temporal hazards using BIM and construction schedules	✓			
Ma et al. (2020)	A deep reinforcement learning approach for robot path planning			✓	
Martinez (2001)	STROBOSCOPE: State and resource-based simulation of construction processes		✓		
Mnih et al. (2015)	Human-level control through deep reinforcement learning	✓			
Mohammadi and Tavakolan (2019)	A hybrid safety risk assessment approach for construction projects		✓		
Nikolic and Ghorbani (2011)	Developing agent-based models based on institutional statements		✓		
Park et al. (2021)	Immersive simulation environments for safety awareness training in construction	✓			✓
Pedro et al. (2016)	Framework for integrating safety into VR-based construction training				✓
Sadeghi et al. (2019)	Linking BIM and ABM for construction safety analysis	✓	✓		

Author(s) & Year	Title	BIM	ABM	RL	Game Engine
Sacks et al. (2009)	Construction safety planning using the safety activity theory model	✓			
Sutton and Barto (2018)	Reinforcement Learning: An introduction			✓	
Tavakolan and Nasirzadeh (2014)	An agent-based model for evaluating construction projects' safety performance		✓		
Teizer and Cheng (2015)	Proactive safety simulation in 4D BIM environments	✓			
Wang and Truijens (2018)	BIM-based immersive visualization for safety training using Unity3D	✓			✓
Wang et al. (2014)	Serious games for workplace safety				✓
Xu et al. (2023)	Examining construction group's safety attitude resilience under major disruptions		✓		
Yang et al. (2023)	A reinforcement learning-enhanced ABM for dynamic construction safety simulation		✓	✓	
Zaman et al. (2024)	Towards an Integrated Framework for Digital Twins in Construction Safety Training	✓			✓
Zhang and Fang (2022)	Behavioural modelling of construction workers		✓		
Zhang et al. (2013)	BIM and safety: Automatic safety checking of construction models and schedules	✓			
Zhang et al. (2015)	Simulation-based evaluation of training effectiveness in construction safety		✓		
Zhao et al. (2022)	Reinforcement learning applications in construction: A review			✓	

2.1. Building Information Modelling (BIM) for Safety Management

Prior to the adoption of Building Information Modelling (BIM), construction safety planning was typically reactive, fragmented and largely based on 2D documentation. Site safety inspectors and supervisors relied on printed hazard logs, safety signage, toolbox talks, and static checklists to manage risk. These methods often proved unfruitful as they suffered from poor coordination between design and field teams. Additionally, these methods limited visualization of complex spatial arrangements as well as created an inability to foresee the dynamic interaction that would occur between workers, equipment and the environment (Teizer et al., 2013; Sacks et al., 2018). Safety data was often soloed, inconsistently updated or entirely absent during the design phases, leading to oversights that manifested during actual construction.

BIM offers a significant advancement that enables real-time collaboration across disciplines by the introduction of a data-rich and model-based process. Rather than representing isolated safety data, BIM directly integrates it into parametric model elements. This allow for the embedding of hazards, risk zones as well as protective systems into the digital twin of the construction environment (Azhar, 2017; Hardin and McCool, 2020). This level of integration supports early detection of safety issues during design, especially when paired with automated rule-checking systems. Studies have demonstrated BIM's utilization in hazard detection, spatial conflict analysis and regulatory compliance. For example, Guo et al. (2012), so as to identify potential collision points and exclusion areas in advance, applied BIM in the simulation of crane operation zones. In a similar fashion, Zhang et al. (2015) developed a model in BIM that flagged incomplete edges and structural voids prior to construction.

The incorporation of safety parameters into BIM workflows allows for pre-construction risk identification and mitigation. Safety managers can simulate spatial relationships between building components, equipment, and temporary works, enabling early detection of conflict zones such as proximity to unprotected edges, inadequate clearance, or workspace congestion (Zhou et al., 2012; Kim et al., 2013). Through integrated safety object libraries, BIM models can embed metadata relating to guardrails, scaffolds, signage, or restricted zones, allowing for rule-based compliance checking using tools like Solibri Model Checker and Autodesk Navisworks.

BIM has also enabled standardisation in safety auditing by encoding regulatory requirements into reusable rule sets. For example, checks for edge protection, barrier placement, and working-at-height constraints can be semi-automated through embedded validation logic (Dirgen Tözer et al., 2024). These features reduce reliance on manual inspection or domain expertise during early design phases and facilitate iterative refinement of safety strategies.

Advanced BIM platforms such as BEXEL Manager and Tekla Structures allow users to annotate and schedule the installation of temporary safety elements, offering support for compliance-based planning. Metadata attached to model components can specify usage conditions, permissible durations, and removal triggers, supporting structured safety planning. Guo et al. (2020) highlight the ability of BIM to streamline coordination of such temporary installations, ensuring alignment between design intent and on-site execution.

Despite these developments, the application of BIM in safety planning has largely remained constrained to static rule sets or compliance verification procedures. Most safety audits within BIM environments are conducted using pre-scripted logic or hard-coded rule libraries, which assume predictable construction progressions and predefined hazard types (Chi, Caldas and Golden, 2014). This makes them less effective in detecting context-specific or emergent risks that arise from changes in sequencing or site layout. For example, temporary slab removals or scaffold dismantling phases may create hazardous voids that remain undetected if not explicitly defined within the rule sets.

As illustrated in Figure 1, traditional BIM-based safety management systems often apply pre-scripted rules at specific construction task transitions. These rules are typically derived from established safety ontologies, regulatory databases such as OSHA, and libraries of industry best practices. When a safety issue is flagged, the system suggests corrective actions from a predefined repository and generates an action report for stakeholder review. While this approach introduces procedural rigour, it remains inherently reactive and limited by the scope of encoded knowledge (Chi, Caldas and Golden, 2014). It does not accommodate unforeseen hazards or deviations from the planned schedule—highlighting the need for more adaptive, behaviour-driven safety modelling.

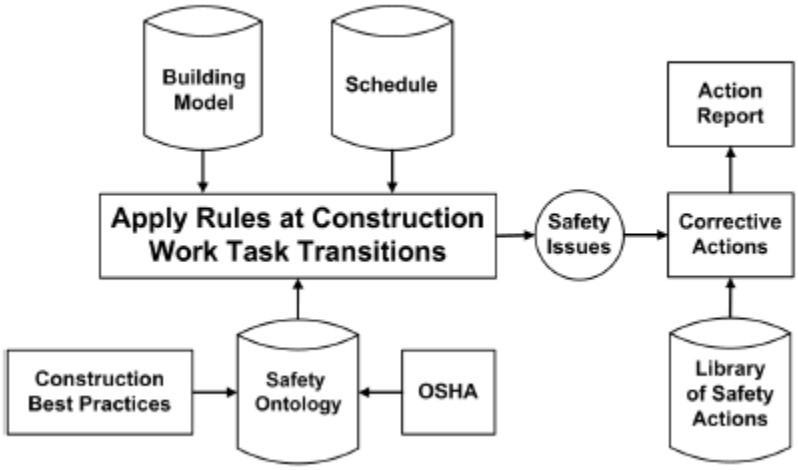

Figure 1 - Framework for implementing an automated rule-based safety checking in BIM (Source: Zhang, 2014)

Additionally, BIM's capacity to simulate behavioural responses or model how workers interact with temporary conditions is inherently limited. While visual walkthroughs may assist in understanding risk zones, they do not inherently support autonomous safety decision-making or adaptive interaction modelling. Interoperability challenges—particularly when integrating third-party safety datasets or custom simulation components—further constrain BIM's potential in fully automating hazard detection (Afsari, Eastman and Shelden, 2021).

Recent efforts have sought to extend the utility of BIM by exporting model data into game engines and immersive simulation environments. These transitions aim to enhance spatial awareness and improve end-user comprehension through real-time interaction. However, such applications still primarily rely on static model geometry and predefined logic rather than enabling adaptive or learning-based agent

interaction (Fang et al., 2020; Alves and Junior, 2020). As such, while BIM offers a robust platform for compliance-based planning and spatial hazard visualisation, its role in simulating dynamic or emergent safety conditions remains limited.

2.2. 4D BIM and Safety Simulation

4D introduces the temporal dimension where 3D components are linked to the construction while the traditional BIM provides the building's physical elements in a static manner. This incorporation enables planners and safety engineers to simulate the constantly evolving workings of a construction site by providing a dynamic lens through which safety risks assessment may be carried out. As noted by Koo and Fischer (2000), 4D models offer momentous potential in construction sequencing, detection of clashes and the visualization of temporary structures such as scaffolding. The changeover from 3D to 4D modelling has been critical in the enhancement of the accuracy and relevance of safety analysis. By aligning safety checks with the construction timeline , 4D BIM permits for the anticipation of hazards that may possibly only emerge at specific stages of construction.

For example, at the onset of the project, hazards such as unguarded voids or incomplete floors may not pose as hazardous, but later transform as such at intermediate stages. Zhang et al. (2013) made an emphasis about this by demonstrating the employment of 4D safety checking. The assessment would be for forecasting when certain protective measures would be needed, using rule-based simulations aligned with the project's Gantt chart.

Additionally, 4D supports the use of what-if analyses in safety management. The application allows for the manipulation of the sequence of activities and construction activities. Safety planners can then simulate multiple alternative scenarios to assist them in the identification of the safest construction path. Tools such as Navisworks and Synchro have been widely adopted to visualise such time-dependent simulations, offering granular control over the scheduling and appearance of temporary safety installations. The usage of these tools has significantly increased as they do not just visualize walkthroughs but also as a basis for automated reasoning and early risk flagging (Lu and Olofsson, 2014; Zhao et al., 2020).

Despite these advantages, the application of 4D BIM to proactive safety simulation remains underdeveloped in practice. Afsari et al. (2021) observed that rather than focusing on behavioural prediction or hazard forecasting, 4D simulations were still mainly used for visual review and clash detection. Similarly, Zhou et al. (2022) found that despite advancements in research related to safety-related 4D modelling, there is still a hindrance to its adoption by a lack of interoperability, user training and integration with active learning systems. Moreover, an additional challenge lies in the representation of temporary risk elements within 4D BIM. It is easy to model permanent structures in BIM authoring tools such as Revit or ArchiCAD. However, temporary site conditions such as edge voids, formwork or material stacks are often modelled informally or completely omitted from the onset, limiting the accuracy of safety simulation. Dirgen Tözer et al. (2024) highlighted the importance of the representation of these temporary elements to avoid blind spots specifically in relation to fall hazard identification.

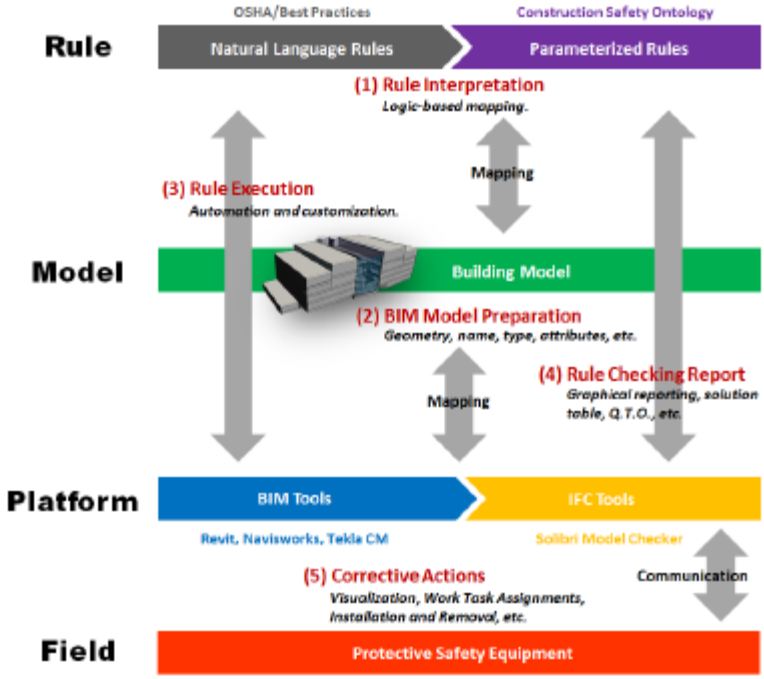

Figure 2 – Multi-level Framework for Safety Rule Enforcement in BIM-Based Environments (Source: Zhang, 2014)

In a bid to overcome the outlined limitations, as previously stated and as illustrated in Figure 1, some researchers, have proposed the use of rule-based extensions and plug-ins specifically related to safety that automatically flag high-risk tasks as the simulation timeline progresses. For example, Zhang et al. (2015) combined BIM and ABM[1] to demonstrate how construction tasks can be coupled with hazard occurrence rules to predict the likelihood of unsafe events. The combination establishes a foundation for more intelligent simulation paradigms.

A particularly illustrative example of this multi-layered pipeline is shown in **Figure 2**, where safety rules are translated into logic-based interpretations and executed via BIM-integrated platforms to identify risks and communicate corrective actions directly to site-level operations. The structured flow demonstrates the potential use of 4D BIM not only as a modelling tool but as a dynamic safety management system embedded into the lifecycle of construction execution (Kim et al., 2013). Merging temporal awareness with model-based reasoning paves way for what can be described as a predictive safety simulation. Instead of the provision of a simple site visualization, the goal converts to foreseeing the time and location of the placement of interventions, ensuring that safety planning is an embodiment of the sequencing logic. This predictive layer is where 4D BIM begins to intersect meaningfully with agent-based and intelligent systems, which are discussed in the following section.

[1] To be detailed further in Section 2.3

2.3. Agent-Based Modelling (ABM) in Construction Safety

Agent-Based Modelling (ABM) is a computational simulation method that represents individual entities, or "agents", operating autonomously within a defined environment according to behavioural rules (Bonabeau, 2002). Agent-Based Modelling (ABM) has emerged as a versatile approach for simulating individual and group behaviour within dynamic and spatially complex systems, such as construction sites. At its core, ABM is based on the representation of autonomous entities, known as agents, which operate according to programmed behavioural rules within a virtual environment (Bonabeau, 2002). These agents may represent construction workers, machinery, vehicles, or other entities interacting on a jobsite. Their local decision-making and interactions with one another allow researchers to observe emergent phenomena such as crowd dynamics, spatial congestion, safety violations, or task interference.

In relation to the construction industry, ABM has gained traction due to its modelling approach where decisions and human behavioural patterns on site are decentralised. This feature aids researchers and practitioners alike in the simulation, analysis and mitigation of hazardous situations before they are actualised. For example, Sacks et al. (2009) developed an agent-based framework to evaluate the spatial coordination of crews during concrete formwork activities. Their simulation revealed potential for task overlap and proximity risks that traditional planning overlooked. Similarly, Zhang et al. (2015) created an ABM to test the effectiveness of safety training interventions by simulating different worker responses to warning signage and supervision levels.

Evacuation scenarios have also been widely explored. Guo and Yiu (2016) developed an ABM to simulate emergency evacuations in high-rise buildings under construction, assessing how various stair configurations and obstruction placements affected egress time and congestion. Their findings provided insight into how real-time site layout influences escape behaviours, especially in constrained environments. In a similar vein, Zohdy, Omar and McCabe (2020) implemented agent-based evacuation modelling on scaffolding platforms, accounting for agent fatigue and speed variations.

Traffic and collision risks between workers and machinery represent another focus area. Liu et al. (2018) created a hybrid ABM to analyse pedestrian–vehicle conflict zones on construction sites, revealing how different scheduling and site logistics strategies impacted the frequency of near-miss events. The ability to test multiple layouts and work schedules using ABM has provided planners with a powerful tool for visualising safety-critical scenarios prior to site implementation.

It should be noted that ABM allows for the exploration of risks that are emergent as they do not arise from unsafe actions that occur in isolation but from the cumulative effect of numerous agents interacting within the environment. Zhang et al. (2020) demonstrated that the intergration of wearable sensor data with ABM frameworks can aid in the simulation of how fatigue or inattention dissaminates among crews which may increase the collective risk exposure. This approcah shits beyond static hazard checklist and embraces a more holisitc view of safety as a dynamic and responsive process. A predominantly persuading application of ABM in safety research is embedded in its capacity to model psychological and behavioural resilience in the aftermath of disruptive incidents. Xu et al. (2023) proposed an ABM framework that would aid in the quantification of the safety attitudes of construction groups. The framework would quantify the deterioration and subsequent recovery of a group following an impactful event such as a fatality or near-miss. Their work presented a two-phase recovery curve that captures

both the immediate psychological response as well as the restoration of safety culture that may occur at a later time. Figure 3 below is particularly pertinent in today's current safety planning practices as it enables project managers and planners to test the effectiveness of various interventions. The interventions such as training, reinforcement or team restructuring were introduced in a bid to shorten the recovery phase or in the minimization of the depth of the initial response shock. Such foresight that is driven by simulations is valuable for the mitigation of risks as well as improving worker morale and ensuring safety engagement over time.

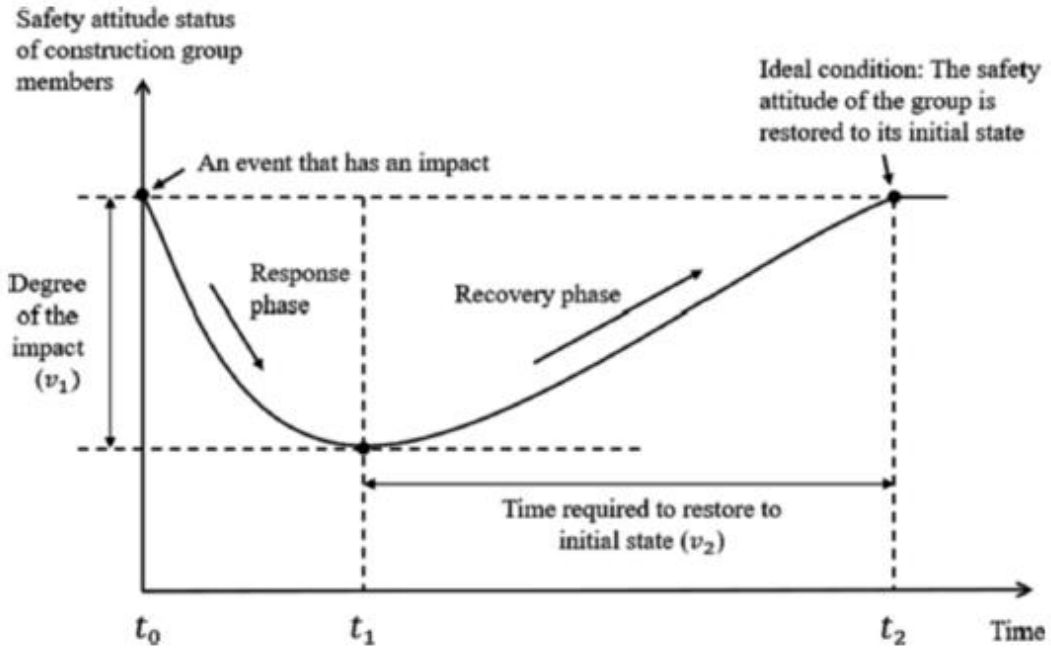

Figure 3 – Construction workers' safety attitude resilience model (Adapted from Xu et al., 2023; originally from Guo et al., 2020).

Furthermore, ABM when used in tandem with 4D BIM, can be used for the visualization of evolving safety scenarios. When coupled with construction sequence data, agents can in synchrony provide simulations of unsafe site behaviour along with construction progress, identification of bottlenecks or zones where congestion frequently occurs (Zhang et al., 2015). The combination of BIM with ABM offers a hybrid enhancement of the predictive capabilities of digital safety planning as it bridges static model data with instantaneous simulation logic. On the other hand, while ABM provides granular insights into behavioural safety, its realism is contingent on the quality of the assumption underpinning each agent's logic. A key challenge persists in the definition of a plausible set of rules applicable for human behaviour especially in the context of cultural, organisational or variables related to stress (Lu & Olofsson, 2014.) Moreover, ABM simulations often require extensive calibration and validation based on field data. This is usually something that is not always readily available in the context of construction. Despite these limitations, ABM holds significant potential in its ability to serve as a complementary layer in the realm of digital safety simulations due to its ability to model collective behaviour, psychological response and human based interaction. These features positively rank its suitability for integration into hazard forecasting and construction safety planning frameworks.

2.4. Reinforcement Learning (RL) in Construction Safety

Reinforcement Learning (RL), a subfield of machine learning, has gained increasing interest in recent years for its potential to simulate adaptive decision-making in dynamic environments. At its core, RL enables an autonomous agent to learn optimal behaviours through repeated interaction with an environment, guided by a system of rewards and penalties (Sutton and Barto, 2018). Unlike rule-based or supervised learning methods, RL does not rely on predefined labels or deterministic scripts. Instead, it continuously refines its strategy (policy) based on trial-and-error exploration and feedback, making it particularly useful in contexts where outcomes are uncertain or environments evolve over time.

The standard RL framework consists of an agent, environment, state space, action space, and a reward function. The agent observes the current state of the environment and selects actions that influence future states. It then receives feedback in the form of rewards (or penalties), which guide future action selection. Over time, the agent seeks to maximise cumulative reward by improving its policy, often through methods such as Q-learning, policy gradients, or actor-critic algorithms (Konda and Tsitsiklis, 2000; Mnih et al., 2015). In simulation contexts, RL is typically implemented within a Markov Decision Process (MDP), where the probability of reaching a future state depends only on the current state and action. This structure supports scalable learning across environments such as robotics, video games, logistics, and increasingly, construction site simulation (Zhao et al., 2022).

Although RL has been widely studied in other engineering disciplines, its integration into construction has been relatively recent. Early works often focused on task scheduling and path planning. For example, Lin and Yang (2014) applied Q-learning to optimise construction crane movements in congested urban sites. Similarly, Khalili and Helander (2020) developed an RL-based planner for tower crane path optimisation, demonstrating improved efficiency compared to rule-based systems.

In terms of construction site navigation and safety, RL's ability to autonomously explore and react to unfamiliar conditions has made it a candidate for simulating mobile agents, such as workers or inspection drones. Ma et al. (2020) employed a deep reinforcement learning approach to train robotic agents for pathfinding in partially obstructed site layouts, simulating avoidance of dynamic obstacles. Their model showed emergent behaviours that reflected collision-avoidant movement, suggesting that RL agents can internalise spatial constraints without being explicitly instructed.

Safety-specific RL studies are still limited but growing. Lee et al. (2022) introduced a reinforcement learning agent for hazard avoidance in simplified BIM environments. Using a reward structure penalising proximity to predefined hazards, their agent learned to navigate around risky areas. Guan et al. (2021) explored an RL-based simulation for fall prevention, in which virtual agents received negative rewards for entering zones flagged as unprotected edges. These studies demonstrate the potential of RL to adapt to complex spatial arrangements and changing site geometries without relying on fixed path rules.

Reinforcement learning has also been tested in virtual reality (VR) and digital twin contexts. Gao et al. (2021) integrated RL with game engine simulations to replicate worker training scenarios, enabling AI-driven feedback loops within immersive environments. This approach supports both skill development

and safety education by dynamically responding to trainee behaviour in real-time. Compared to ABM and rule-based logic, RL presents several key advantages for simulating safety-related behaviours:

- **Adaptivity**: RL agents can adjust strategies dynamically based on environmental changes, without the need to hardcode new rules.
- **Generalisation**: Once trained, agents can often perform across varying layouts or unseen environments, enhancing scalability (Zhao et al., 2022).
- **Policy Optimisation**: RL systems inherently aim to optimise long-term outcomes (e.g., reduced hazard exposure or improved exploration), rather than relying on local or immediate decisions.
- **Stochastic Resilience**: RL models can account for uncertainty and randomness in agent decisions, making them more reflective of real-world worker variation.

These properties make RL a promising tool for modelling safety behaviours that are situational, context-dependent, and difficult to predefine.

Despite these strengths, RL applications in construction safety remain in their infancy. Several practical and methodological challenges persist. One major constraint is the complexity of environment modelling. RL agents require thousands or even millions of interactions to learn effective policies, necessitating detailed and responsive virtual environments (Zhao et al., 2022). The creation of such environments—especially ones that accurately reflect partial construction states, evolving site geometry, or hazard metadata—is both time- and data-intensive.

Another issue lies in reward function design. Defining what constitutes “safe” or “unsafe” behaviour in a quantitative reward structure is not trivial. Sparse or misleading rewards can lead to suboptimal learning or unsafe exploration (Gao et al., 2021). Furthermore, without real-world data or physical measurements for calibration, RL agents may develop behaviours that are mathematically optimal but not necessarily aligned with human-safe practices.

Interpretability also remains a barrier to industry adoption. Unlike rule-based systems, where decision logic is transparent, RL policies, especially those based on deep neural networks, are often “black-box” in nature. This can reduce stakeholder confidence, particularly in critical safety scenarios where human oversight is essential (Amodei et al., 2016).

From a technical standpoint, transfer learning, the ability of an RL agent trained in one environment to operate in another, has shown promise but remains underdeveloped in construction applications. Most existing implementations are limited to fixed environments, requiring retraining or manual adjustment to new sites (Guan et al., 2021).

Finally, ethical considerations surrounding simulated unsafe behaviour, especially in training scenarios, require careful framing. Rewarding hazardous exploration in simulation must be balanced with a clear distinction from encouraging risky behaviour in real-world settings (Leike et al., 2017).

2.5. Game Engines in Construction Safety Simulation

The use of game engines in construction research has gained considerable traction due to their capacity for real-time 3D visualisation, physics-based simulation, and agent interactivity. Originally developed for entertainment and gaming, engines such as Unity, Unreal Engine, and Godot have since been repurposed as powerful platforms for immersive training, behavioural simulation, and interactive visualisation across engineering domains (Wang et al., 2014; Afsari, Eastman and Shelden, 2021). In the construction sector, these engines enable the development of digital environments that replicate site conditions with high spatial and temporal fidelity, thereby supporting proactive safety planning, educational modules, and AI-based experimentation.

Modern game engines offer a combination of rendering pipelines, physics systems, animation controllers, and scripting APIs, allowing for flexible and realistic digital twin environments. This flexibility is essential in construction contexts, where the simulated environment must respond dynamically to agent behaviour, structural progression, and hazard emergence (Wang and Truijens, 2018). Game engines also support collider-based detection, navigation meshes, and real-time lighting that enable interactive experiences grounded in physical logic. When integrated with BIM-derived geometry, these features allow for accurate spatial feedback, such as detecting collisions with ledges, interactions with scaffolding, or movement along staircases, during virtual construction walkthroughs or training simulations (Zhang, Chi and Lee, 2021).

Unity and Unreal Engine are the most widely adopted platforms in construction simulation studies due to their extensive documentation, cross-platform support, and community-developed libraries. Unity, in particular, has been employed in a variety of safety training contexts. For instance, Pedro et al. (2016) developed a Unity-based virtual reality (VR) module to train workers in the identification of fall hazards and unsafe site practices. Participants could navigate a virtual site environment using head-mounted displays and receive instant feedback based on their decisions. Similarly, Chan et al. (2021) used Unreal Engine to simulate confined-space hazards and test evacuation strategies. The interactive nature of the environment allowed researchers to assess behavioural responses under timed and constrained conditions, offering insights not easily captured by static BIM visualisations. Both engines also support integration with reinforcement learning toolkits such as ML-Agents (Unity) and OpenAI Gym (via Python bindings), facilitating the training of adaptive AI agents within construction-like environments (Gao et al., 2021). This capability has opened new pathways for simulating safety-aware behaviours that evolve over time.

Godot Engine, though newer and comparatively less adopted in construction literature, presents a robust open-source alternative. It offers built-in scripting with GDScript (or C#), scene management tools, and an active developer community. Its lightweight architecture makes it particularly suitable for academic prototyping, enabling users to deploy 3D simulations without licensing constraints or large memory overheads (Alves and Junior, 2020). While examples of Godot in mainstream construction research remain limited, recent work has explored its potential as a simulation host for training intelligent agents in built environments. Studies have reported its successful use in architecture and planning education due to its modular scene graph and ease of importing IFC-derived 3D assets (Diniz et al., 2022). Moreover, its support for raycasting, custom physics layers, and agent kinematics allows researchers to script behavioural experiments with precision, a key requirement in safety-critical applications. The

emergence of Godot 4.x has also enhanced the engine's compatibility with external AI frameworks, enabling it to serve as an environment for reinforcement learning via Python-Godot bridges or WebSocket integrations. These features suggest a growing role for Godot in experimental construction simulations, particularly in academia and open-source research settings.

Compared to traditional BIM viewers or rule-based simulators, game engines offer the following advantages:

- **High-Fidelity Spatial Representation**: Allows users to explore the virtual site in first-person or third-person view with dynamic lighting, shadows, and material realism.
- **Real-Time Physics and Behavioural Feedback**: Enables agents to interact with moving platforms, fall from ledges, or respond to changing gravity or friction conditions.
- **Customisable Reward Structures and Logging**: Essential for reinforcement learning or behaviour logging during hazard-seeking or navigation tasks.
- **Immersive Training and Stakeholder Engagement**: Supports VR headsets, haptic controllers, and interactive interfaces for training, walkthroughs, and stakeholder review.

These features collectively transform passive BIM models into active, behavioural environments that are responsive to user input and agent logic, offering a deeper understanding of spatial safety conditions.

Despite their versatility, game engines also introduce challenges. One common issue is the translation of BIM data into a format usable by game engines. While IFC files can be imported into intermediate platforms like Blender, the process often results in loss of metadata or requires remapping of materials and hierarchies (Afsari et al., 2017). Additionally, game engines are not natively designed to support construction-specific ontologies or scheduling data, requiring custom scripts or plugins for 4D integration.

Another limitation concerns the validation of behavioural realism. While game engines allow agents to simulate movement and interaction, the fidelity of their decisions depends heavily on the physics and AI logic embedded within the engine. Without appropriate calibration, simulated behaviours may diverge from real-world expectations, especially in safety-critical tasks (Zhao et al., 2022). Lastly, the interoperability between simulation platforms, data storage systems, and analysis frameworks remains a technical hurdle. Ensuring seamless communication between BIM models, training logs, and RL algorithms often demands extensive middleware or custom integration efforts.

2.6. Summary of Gaps in Existing Literature

The literature reviewed in this chapter demonstrates significant advances in digital approaches to construction safety simulation, yet several key limitations persist across the examined domains.

In the case of Building Information Modelling (BIM), most implementations remain focused on static hazard visualisation and rule-based assessments. Despite the availability of 4D scheduling tools, their

application in safety planning tends to be compliance-oriented, with limited support for modelling the evolution of risk throughout the construction sequence. Integration challenges between BIM software and simulation environments also hinder the seamless translation of geometric and semantic data.

Agent-Based Modelling (ABM), while valuable for representing worker-environment interactions, continues to rely heavily on predefined behavioural scripts. This constraint limits its effectiveness in simulating adaptive or unanticipated safety responses, particularly in dynamic and partially completed site conditions.

Reinforcement Learning (RL) has shown promise in adjacent domains, but its application in construction safety remains nascent. Existing studies have not fully explored how learning-based agents might autonomously identify and react to site hazards that are emergent, incomplete, or unlabelled. The integration of RL with spatially complex and evolving construction environments is also underdeveloped.

Lastly, while game engines provide the technical capacity for real-time simulation and behavioural testing, their use in safety-focused research has been limited. Most applications to date have prioritised user-controlled experiences or visual walkthroughs, rather than autonomous simulations with learning agents. Interoperability issues between BIM models and game engines further complicate attempts to establish coherent simulation workflows.

Together, these limitations point to a fragmented landscape in which behavioural simulation, safety modelling, and digital construction tools have yet to be fully integrated into a cohesive, adaptive framework.

3. METHODOLOGY

The objective of this chapter is to provide a detailed methodological account of how the proposed simulation framework was designed, implemented and evaluated. In line with the exploratory nature of the research, the methodology focuses on the technical feasibility of integrating Building Information Modelling (BIM), game environments and Reinforcement Learning (RL) agents for proactive construction safety analysis. Rather than pursuing performance optimisation or benchmarking against industry metrics, the methodological deign emphasises modular integration and proof of concept validation. This aligns with observed gaps in the literature, where existing studies often treat BIM, simulation and intelligent agent logic as disconnected components (Zhou et al., 2022; Duan, 2025).

This methodology is structured around a sequential pipeline comprising five stages as outlined below.

1. BIM and asset preparation,
2. Environment setup within a game engine,
3. RL agent definition and configuration,
4. Training execution and;
5. Output logging and visual analysis.

These stages were derived from existing practices in digital construction (e.g., Park et al., 2021; Fang et al., 2020) but adapted to suit the unique demands of simulating autonomous hazard-seeking behaviour within partially constructed 4D BIM environments. Each stage is elaborated with specific tools, formats and logic structures used in implementation, such as the translation of IFC files from Revit into Blender to reduce mesh complexity while retaining structural fidelity. Within Godot, spatial logic is implemented through a combination of nodes and the setup of physical bodies and ray-based sensors. The training of the agent uses proximal policy optimisation (PPO) from the stable_baselines3 python library. The training and evaluation pipeline is entirely contained within the simulation environment, allowing agents to interact with spatial data in real time without external data dependency.

The methods described in this chapter serve to illustrate technical feasibility as well as to demonstrate how such systems can be constructed and scaled for future use. Extra mind is paid to ensure that each methodological choice reflects the constraints and intentions of the research. Emphasis is particularly set on the discovery rather than the avoidance of construction hazards during planning phases. This sets the stage for the discussion of results in the following chapter, which interprets the agent's performance in relation to the design goals established here.

3.1. Research Design

This research adopts an exploratory, design-based methodology with the primary objective of developing and demonstrating a simulation framework capable of integrating Building Information Modelling (BIM), game engine environments and reinforcement learning (RL). The study does not attempt to optimise learning performance or generalise across different construction contexts. Instead,

it presents a proof-of-concept prototype to show that such integration is feasible and potentially beneficial for proactive construction safety analysis.

Unlike traditional safety research grounded in statistical analysis or site-based case studies (Hinze, 2006), this approach relies on simulation and synthetic data generation to mimic unsafe conditions. It aligns with broader trends in construction informatics, where digital twins and virtual environments are increasingly used in the testing and validation of workflows prior to physical execution (Zhou et al., Guo et al., 2020). The chosen strategy centres on constructive realism, where simulation serves as a controlled testbed to explore agent behaviour under varying spatial and hazard conditions. The realism stated is operational rather than physical. This means that hazards are not explicitly tagged or predefined and interaction occurs in ways that are consistent with how safety risks manifest in partially complete buildings (Amer et al., 2023).

Given the novel intersection of technologies involved such as BIM, physics-based simulation and RL agents, this study adopts a pipeline-oriented design. The stages are arranged sequentially to reflect a logical flow. The asset is first prepared, followed by the setup of the environment, agent configuration and training and concluding with the evaluation of the output. Each stage is discrete yet interdependent, while ensuring that methodological transparency is maintained and future feasibility is supported as well.

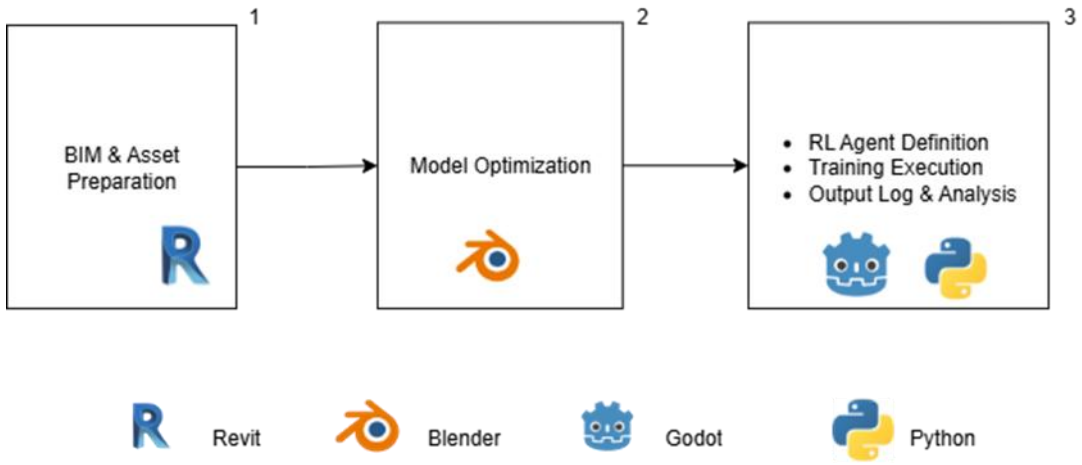

Figure 4 – Methodology sequence

Furthermore, Godot Engine, an open-source 3D game development platform, was chosen for this research as it ensures flexibility and adaptability. Unlike commercial platforms such as Unity or Unreal Engine, Godot allows full access to low-level physics and scripting controls, which is particularly important for simulating sensor-based perception and instantaneous interactions with incomplete or hazardous environments (Fang et al., 2020; Afsari et al., 2021). This design decision supports the broader objective of creating a configurable and extensible testbed for future RL research in digital construction safety.

Importantly, the research does not seek to optimise RL parameters, or train agents ready for production. The core contribution is to demonstrate how a simulation compatible with RL can be built from standard BIM tools and how agents can be trained to explore and interact with simulated hazards. The findings,

rather than deliver safety algorithms ready for deployment, are thus intend to inform planning at early stages, safety reviews and future automation strategies.

3.2. BIM Model Preparation and Export

The initial stage of the methodology involved the preparation and export of a 4D BIM model from a commonly used construction design tool (Revit), followed by its conversion into a format suitable for use in instantaneous simulation. This process was central to enabling the transition from static construction data to a dynamic, navigable virtual environment where reinforcement learning (RL) agents could perceive and interact with hazards. The exported model, representing the partial stage of construction forms the foundation of the simulation and dictates the realism and spatial logic of the environment.

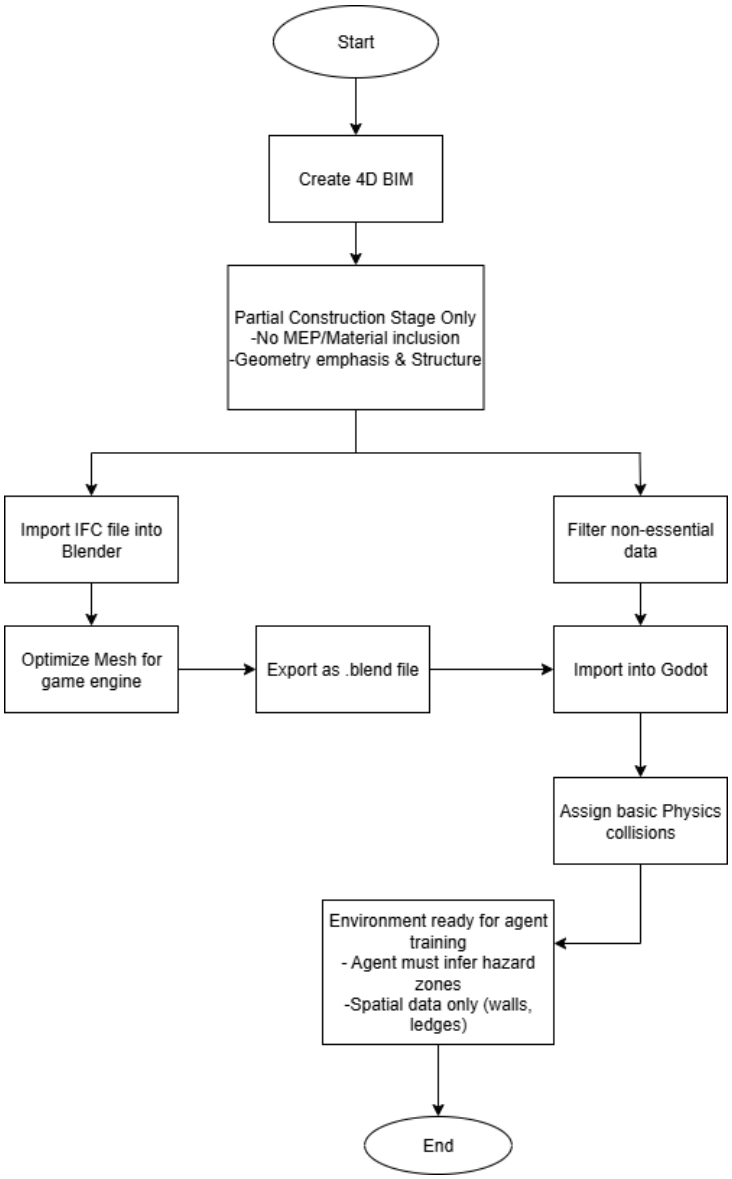

Figure 5 – BIM model preparation and export workflow

Revit was chosen due to its industry-wide adoption and compatibility with open standards such as Industry Foundation Classes (IFC). The source model was developed to reflect a partially constructed multi-storey building with conditions associated with early-stage construction safety risks such as exposed stairwells and open elevator shafts (Zhou et al., 2012; Dirgen Tözer et al., 2024). Only geometric and spatial information was retained, with material and mechanical detail intentionally excluded to optimise simulation performance.

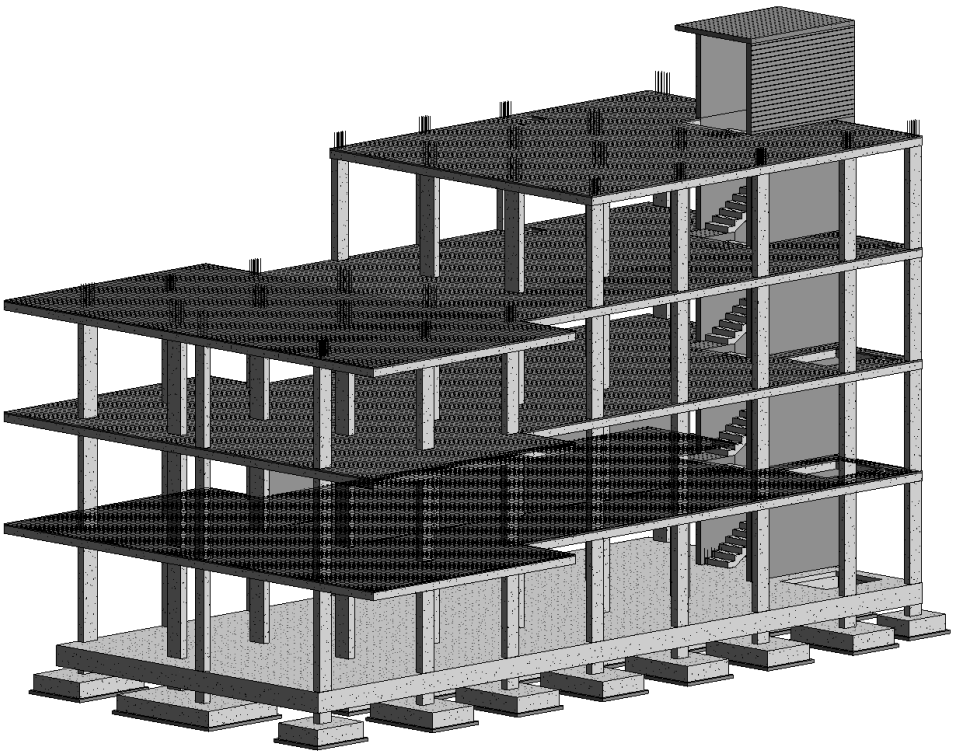

Figure 6 - BIM model in Revit

The model was exported from Revit in the IFC format, which supports open data exchange and preserves essential hierarchical and spatial relationships between building components (Borrmann et al., 2009). This format ensured that key structural elements such as slabs, beams and walls were recognised in downstream tools. Given the limitations of the IFC schema for certain game-engine applications, only elements relevant to spatial navigation and hazard interaction were preserved.

Blender served as an intermediary step between the BIM authoring tool and the game engine environment. The IFC model was imported and several pre-processing tasks were conducted. The optimisation ensured that the hierarchy preservation to maintain object naming and relationships occurs. Such optimisation steps were essential to ensure compatibility with the Godot engine, which, although powerful, has performance constraints when managing highly detailed architectural meshes.

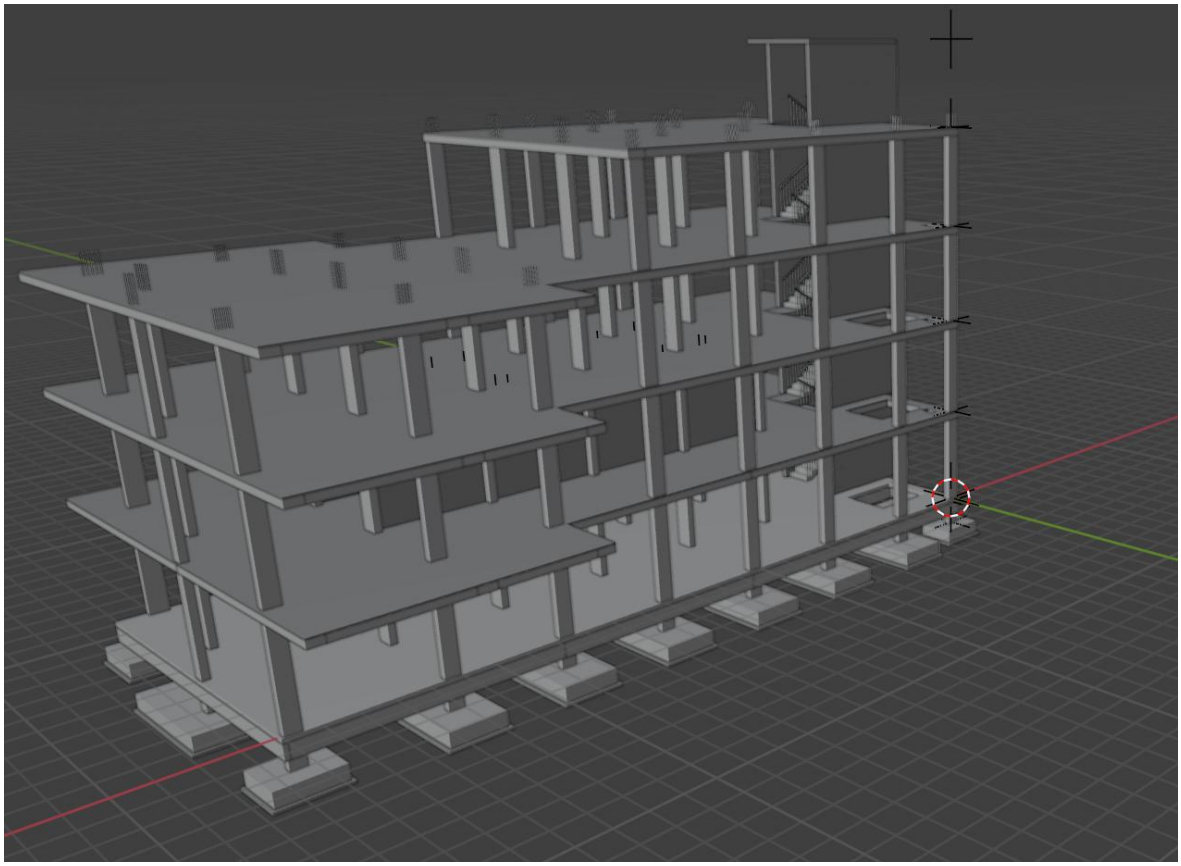

Figure 7 – Optimised BIM model in Blender

The optimised model was exported as a .blend file, Godot's natively supported format. This approach bypassed the need for additional asset conversion or loss of geometry metadata. The .blend file retained object names, mesh origin points as well as the physical scale, which were later used in the assignment of collision shapes. The export pipeline thus established a seamless bridge from authoring tools to the simulation space, allowing the model to transition from a design artefact to an environment that was instantaneous and easily navigable by an RL agent. This process builds upon existing methods for BIM to game engine conversion (Park et al., 2021; Zhao et al., 2020) but refines them for the specific purpose of RL training and hazard discovery.

3.3. Game Engine Environment Setup

The simulation environment serves as the core stage upon which the reinforcement learning (RL) agent interacts with and learns from a partially constructed 4D BIM model. Rather than relying in predefined hazard markers or manually labelled danger zones, this study emphasized autonomous hazard discovery. This means that the agent must infer risks solely based on sensor feedback and environmental consequences. The environment was thus constructed to realistically reflect the incomplete and variable conditions of construction sites, while remaining unannotated and open-ended to support exploratory learning. The choice of Godot Engine as the game platform was due to its open-source nature, flexibility and lightweight performance, making it suitable for both high-frequency physics simulation and custom RL integration.

3.3.1. Importing the BIM model into Godot

The environment design began with the import of the geometry from Blender into Godot. Assets exported from the partial 4D BIM model were imported as .blend files, maintaining object hierarchies, material assignments and spatial coordinates. Each imported mesh was instantiated as a StaticBody3D with attached CollisionShape3D components, enabling physics-based interaction with the RL agent. Unlike traditional BIM to simulation conversions, no semantic tagging or region-specific annotations were applied during import. This ensured that architectural features such as slab edges, staircases, lifts shafts or incomplete floors retained their raw geometric identities, requiring the agent to learn hazard significance through interaction rather than inference from labels. Godot's physics engine operates in real time with gravity, collision and friction models activated. All imported bodies were set to interact naturally with the agent, which meant that potential falls, impacts or environmental transversals occurred as they would in a physically realistic virtual space.

3.3.2. Realism through structural incompleteness

To enable hazard discovery without bias, the model was purposefully selected to represent an incomplete construction phase, as is typical in the early stages of planning, Features that represented latent risks included:

- Unfinished edges of slabs without parapets or barriers
- Large vertical shafts such as stair voids or service shafts
- Irregular floor segments with cantilevers or drops
- Exposed multi-level floor transitions.

These elements were not altered or enhanced with visual warnings or navigation cures. Their representation was entirely geometric and physical, allowing the agent to experience the consequences of unsafe navigation (e.g., falling from a height it entering a shaft) and adapt to its behaviour accordingly.

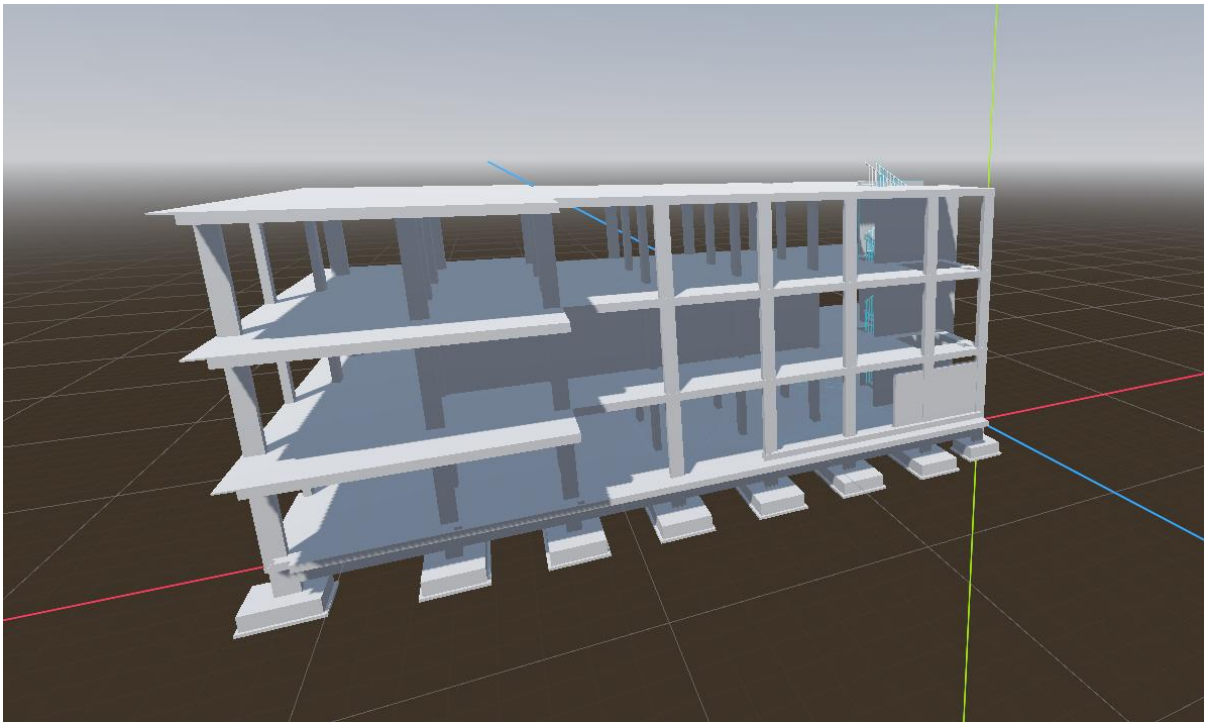

Figure 8– BIM model in Godot with some elements hidden

This formulation of simulation aligns with the concept of "geometry-induced risk" discussed by Zhang et al. (2015) and Kim et al. (2013), where architectural conditions inherently contain danger even if they are not explicitly marked. By refraining from using metadata or markers, this study further diverges from prior work that assumed hazard zones to be a certainty and instead promotes bottom-up learning of spatial risk.

3.3.3. Sensor-based perception design

The agent's understanding of the environment relied entirely on its sensor array, built using Godot's raycasting functionality. The custom RayCastSensor3D module emitted multiple rays from the agent's body in a semi-circular arc that faced forward and downward. Key features included:

- Obstacle detection: Forward-facing rays measured distances to nearby static bodies (walls, columns etc.) enabling obstacle avoidance learning.
- Ledge identification: Downward rays projected near the agent's feet could detect the absence of floor geometry, thus identifying potential falls or voids as is shown in Figure 9.

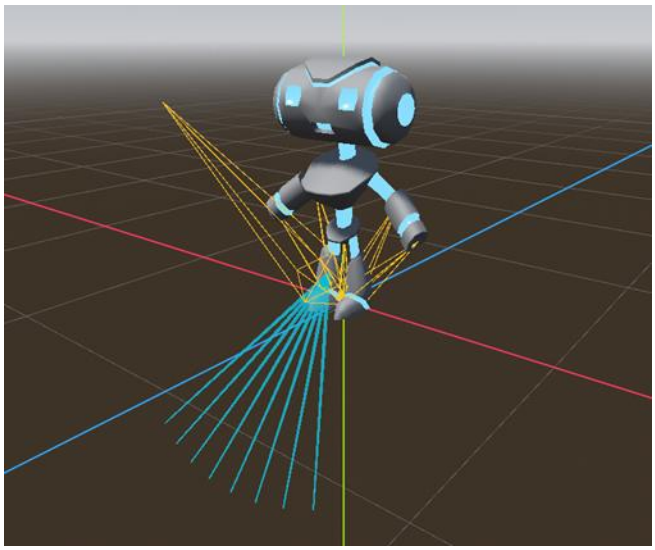

Figure 9 - Rays to detect the absence of floor geometry

- Stair detection: By comparing raycast elevation returns, the agent could potentially detect transitions indicative of stairs or ramps.

Each ray (as displayed in Figure 10) returned either a hit Boolean or a normalised scalar value representing the distance to contact and these were compiled into the agent's observation space per time step. Importantly, no direct hazard status was conveyed, the agent was never told whether something was dangerous or not. Instead, it received rewards or penalties from the environment based on the outcome of its actions, reinforcing the unsupervised hazard-seeking behaviour.

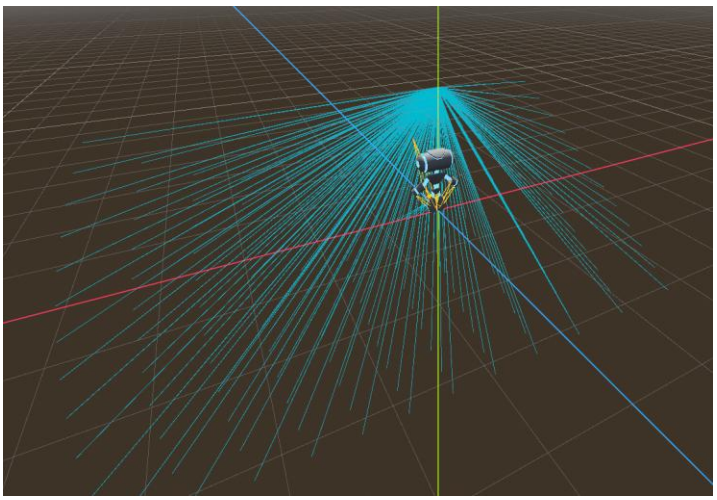

Figure 10 – Rays to detect and measure distances

This approach follows similar principles used in RL systems that were aware of obstacles (Hsu et al., 2021; Guan et al., 2021), but with an inverse logic. The agent was not trained to avoid obstacles but to instead seek and explore potentially unsafe areas in order to simulate safety inspections.

3.3.4. Spawn point randomisation without semantic predefinition

To facilitate robust policy learning, agent spawn positions were randomised at the start of each episode using a custom routine within the Player script. Instead of relying on predefined Marker3D nodes, a curated list of valid (x, y, z) coordinates was hardcoded and maintained within the environment. These coordinates were manually sampled across different floors, spatial zones and orientations to ensure broad environmental coverage while avoiding invalid or obstructed spawn points that would result in immediate failure (e.g., mid-air spawns or atop narrow ledges). This filtering process ensured that all spawn zones were technically valid and physical accessible, while still retaining the semantic neutrality. The workflow is illustrated below.

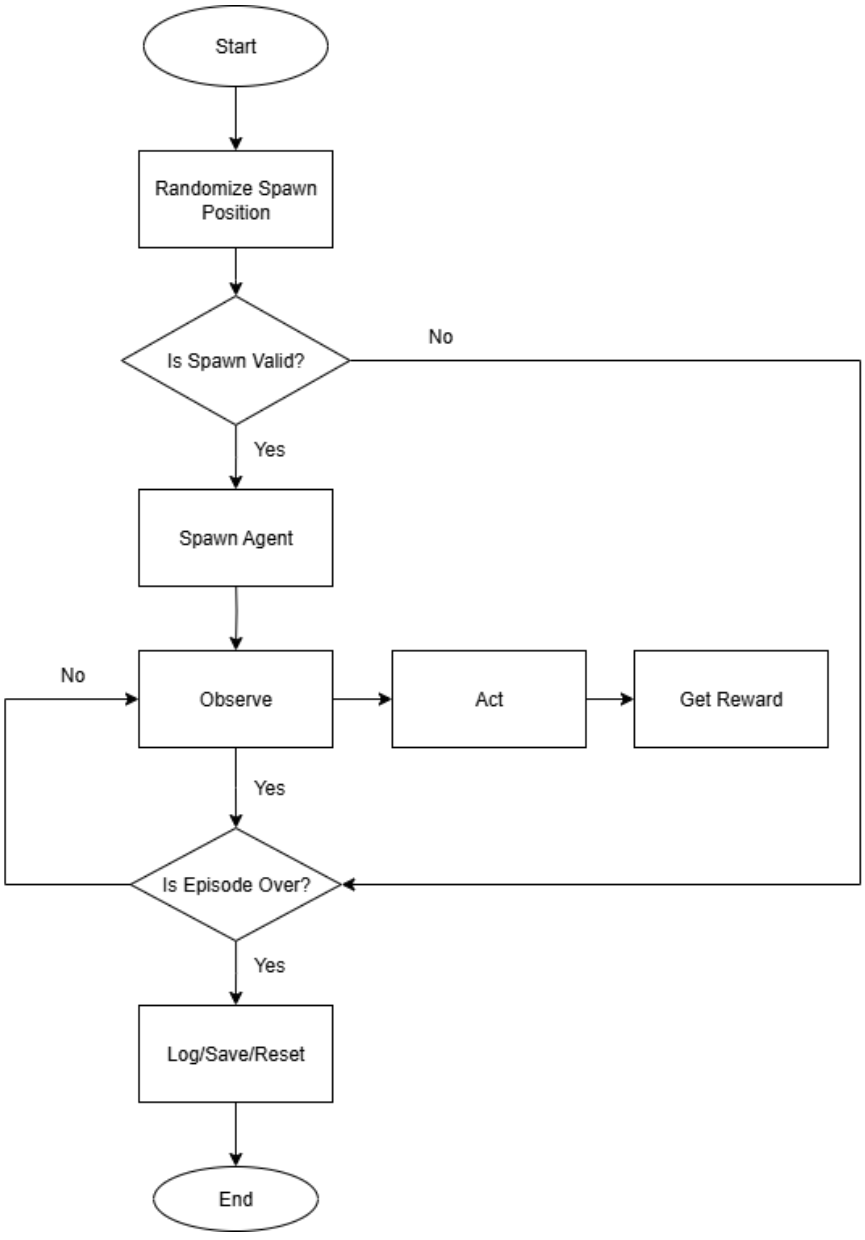

Figure 11 - Spawn point randomisation workflow

Unlike typical procedural simulations where spawn points are tailored toward specific goals or scenarios, the selected coordinates carried no semantic labels or predefined proximity to risk. The agent remained oblivious to its relative distance to hazard or reward elements at the beginning of each episode. This neutral randomisation strategy encouraged exploration and discourages the memorization of fixed danger zones or safe paths.

By keeping the spawn logic unbiased and decoupled from known hazard locations, the design adhered to foundational reinforcement learning principles, particularly in reference to the requirement that the learned policy should generalise across states rather than exploit static spatial patterns (Sutton and Barto, 2018).

3.4. RL Agent setup and configuration

The reinforcement learning (RL) agent served as the autonomous system tasked with the making of decisions within the virtual construction environment. It was required to explore its surroundings and infer the presence of hazards through physical interaction. Unlike traditional safety simulations that rely on following a scripted path or danger zones outlined from the outset, this study adopted a learning-centric approach. The agent learned to identify, approach and interact with hazards based on a trial and error basis. This section describes the architecture, observation space, action design and reward system used to train the RL agent, highlighting how each component was configured to support proactive hazard exploration.

3.4.1. Selection of RL algorithm

The algorithm selected for this study was Proximal Policy Optimization (PPO), a widely used policy gradient method implemented via the stable_baselines3 library in Python. PPO strikes a balance between performance and stability by limiting policy updates through a clipped objective function (Schulman et al., 2017). It is especially well-suited for continuous control problems in high-dimensional state space, such as 3D navigation tasks. PPO was chosen due to its successful application in similar tasks involving robotic navigation (Guan et al., 2021), safe exploration (Duan, 2025) and adaptive policy learning under sparse reward conditions.

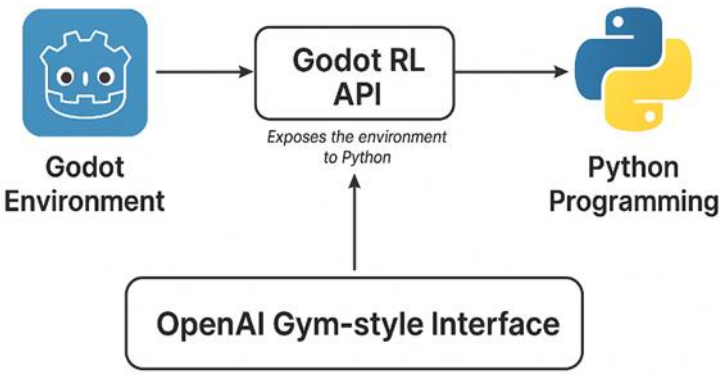

Figure 12 - Interface between Godot and Python

The training was conducted using the Godot RL API, which exposed the game environment to Python for instantaneous interaction via an OpenAI Gym-style interface. This setup allowed the RL agent to receive observations, select actions and receive scalar rewards in synchronised timesteps.

3.4.2. Agent observation space

To interact effectively with the environment, the agent acquired an observation space that was rich and aware of the context. Each observation vector provided to the policy network at every timestep included:

- Sensor Ray Data: Normalised distance readings from a multi-raycast system projecting forward and downward from the agent's body (as detailed in Section 3.3.3). These encoded both obstacle proximity and ledge awareness.
- Velocity Vectors: The agent's movement vector in local space (x, y, z), allowing it to track its own speed and direction.
- Grounded State: A Boolean indicating whether the agent was in contact with a surface, used to detect falls or jumps.
- Floor Height Index: A numerical representation of the agent's vertical position, coarse-grained into discrete levels (e.g., ground floor = 0, first floor = 1, etc.) to support generalisation across multi-storey environments.

No visual inputs (e.g., RGB images) were used to maintain computational efficiency and interpretability.

3.4.3. Action space and control design

The agent was configures to use a continuous control scheme with two core action dimensions:

- Movement Control: Forward/backward movement, represented as a scalar in the range [-1, 1], where -1 signified reverse and 1 full forward speed.
- Turning Control: Left/right rotation, represented as a scalar in the range [-1, 1], mapping to yaw rotation speed.

This design enabled smooth and fluid navigation behaviour, allowing the agent to adaptively steer, reverse or explore tight spaces such as stairwells.

The simplicity of the control scheme was intentional as it minimised the complexity of the action space, allowing learning to focus on spatial awareness and hazard-seeking rather than fine-grained locomotion.

3.4.4. Reward Shaping

Unlike traditional reinforcement learning models that penalize unsafe behaviour, this study reverses the paradigm. The goal is not to train an agent to avoid hazards, but instead intentionally seek out, identify and interact with them. The agent acts as a proactive safety inspector within a simulated construction environment, helping uncover unsafe conditions that might otherwise go unnoticed during early planning stages.

The reward structure reflects this discovery-based philosophy:

- +1.0 to +3.0 reward for falling from an open edge or void, with high falls earning proportionally greater rewards (e.g., a fall from the third floor yields more than a fall from the first). This models the severity and significance of undetected risks.
- +1.5 reward for descending stairs, as this indicated the agent's capacity to detect and utilise vertical circulation routes which is important for assessing multi-level safety planning.
- +0.5 reward for physically entering or interacting with open edges, unguarded stairwells or incomplete slabs. Even if the agent does not fall, this represents early identification of safety gaps.
- -0.2 penalty for becoming stuck (minimal movement for a sustained period), to discourage idle or ineffective behaviour.
- 0.0 neutral reward for general movement or non-hazardous navigation.

This reward system encourages the agent to learn through physical outcomes such as falling into a shaft or walking off an unguarded edge becomes a positive event in training. From a safety planning perspective, these interactions simulate the process of exposing weaknesses in site layout or incomplete construction stages, allowing designers or planners to revise models before practical execution.

This approach aligns with the inverse safety training paradigms, where the agent's failure is reframed as a signal for design correction and not behavioural error. The model therefore promotes early detection of hazardous features in the digital twin of the site, which is especially relevant and of priority in the context of construction models that are evolving over time.

3.4.5. Episode Design and Reset Conditions

Each training episode began with a randomised spawn (Section 3.3.4) and lasted until one of the following conditions was met:

- The agent encountered a predefined number of hazard interactions
- The agent was stuck or non-progressive for more than a stipulated duration
- A maximum time limit was reached

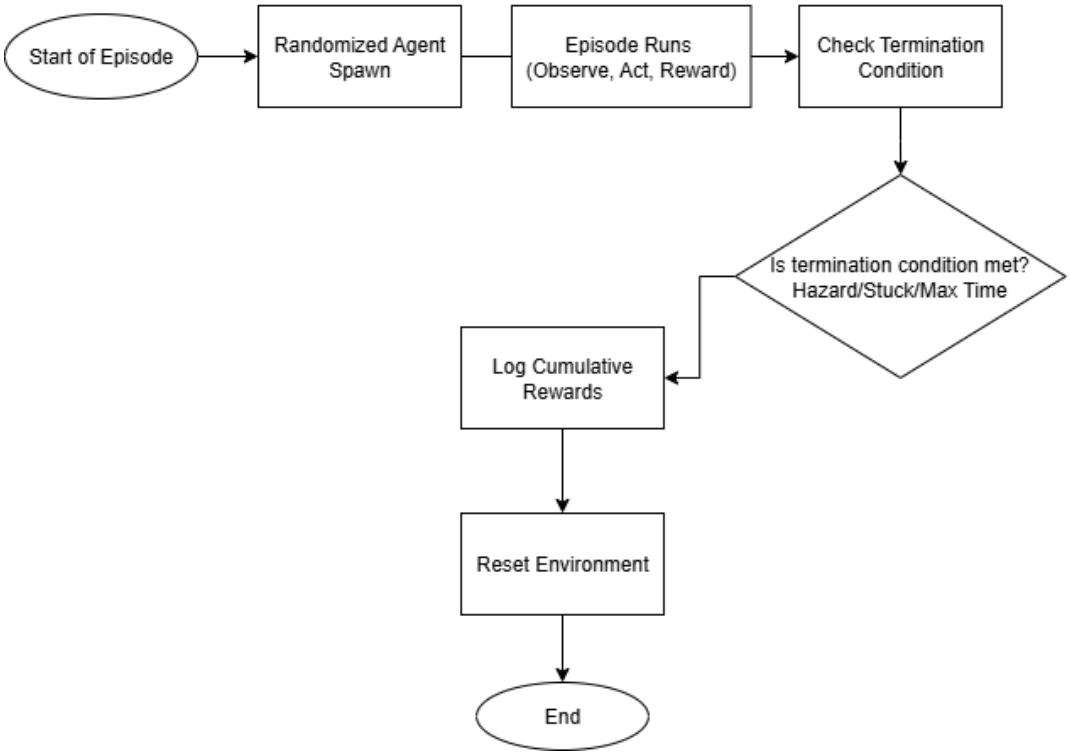

Figure 13 – Episode reset workflow

At the end of each episode, cumulative rewards were logged and the environment was reset. This episodic structure promoted short, focused learning cycles and discouraged overfitting to specific spatial configurations.

3.5. Training loop and convergence process

This section outlines the structure and components of the training procedure used to guide the agent in learning behaviours related to hazards. The aim was to configure a repeatable and scalable training loop capable of processing large volumes of interaction data while preserving architectural and algorithmic consistency across training runs.

3.5.1. Episode lifecycle and environment reset

Training was structured around discrete episodes, each representing a bounded interval in which the agent interacted with the environment under its current policy. At the start of each episode:

- The environment was randomised with the agent spawned at a new location across one of the multiple building levels.
- Previously defined hazards (open edges, voids, shafts) were preserved and no predefined waypoints or navigation cues were included.

Each episode terminated upon satisfying one of the following:

- A predefined number of steps elapsed (e.g., 1000 steps)
- The agent exceeded a "stuck" threshold (minimal movement over time)
- Hazard interaction conditions were logged as triggers (e.g., falling off an edge)

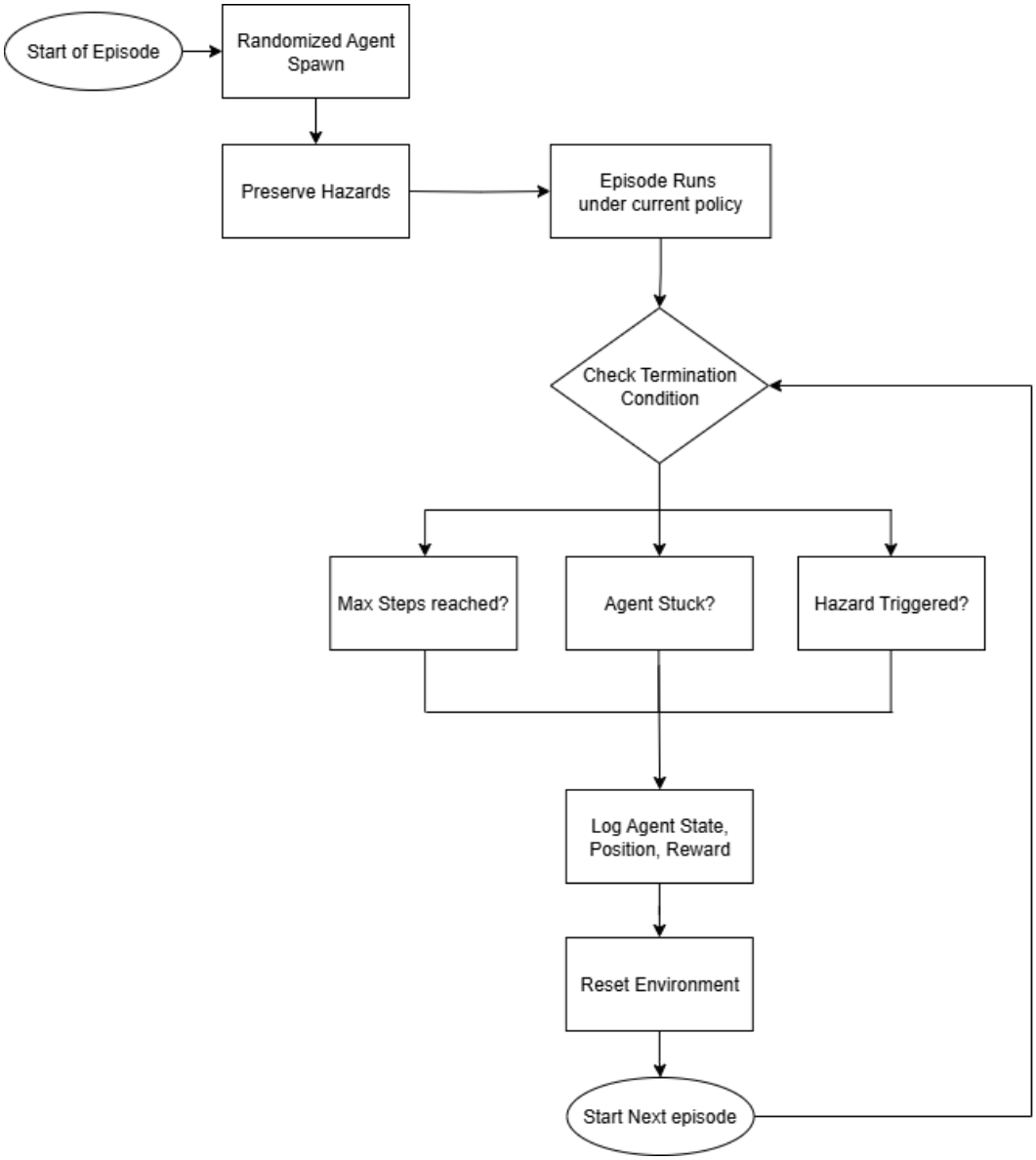

Figure 14– Episode lifecycle and environment reset workflow

After each episode, the environment was reset and key interaction data (such as agent position, state vector and reward signals) were recorded and passed in to the learning algorithm.

3.5.2. Training framework and algorithm configuration

The training pipeline combined external RL libraries with the Godot game engine to provide a modular simulation-training architecture. Key components include:

- Godot Engine (v4.4): Provided the 3D simulation environment with physics, collisions and agent control.
- Godot RL API: Used to expose simulation parameters and step functions to external Python-based training logic
- Stable_baselines3 PPO: Selected for its robustness in continuous state-action spaces and compatibility with non-visual observations.

3.6. Simulation outputs and evaluation

After completing the training phase, the final policy checkpoint was reloaded into the Godot simulation environment for episode replay and basic video capture. These replays were conducted under the same environmental conditions, geometry, and physics setup as those used during training. No changes were made to the agent's structure, sensor logic, or control functions. The purpose of this step was to manually observe the agent's movements across multiple floors of the partially constructed model and to record episodes for later inspection.

Each simulation run was initiated by randomly spawning the agent at one of the predefined valid starting positions within the environment. The agent's movement and interactions were governed entirely by the trained policy, with no further learning or exploration noise applied during this stage. A fixed third-person camera setup was used to follow the agent throughout each episode.

Replay sessions were recorded using screen-capture software. These recordings captured the full simulation viewport, allowing for visual documentation of the agent's movements, collisions, and interactions within the scene. No in-engine visualisation tools such as trail renderers or overlays were implemented. Similarly, structured data logging (e.g., automated export of rewards, positions, or sensor values to .csv or .json) was not configured during this phase. All review and interpretation of the recorded behaviour were conducted manually using the visual recordings, and no real-time quantitative data was exported directly from the Godot engine.

The purpose of these recordings was to support the qualitative assessment of the trained agent's navigation across the 4D BIM environment, which is discussed in Chapter 4.

This page is intentionally left blank

4. RL TRAINING AND RESULTS

The results of the simulation-based reinforcement learning (RL) experiment, evaluating the trained agent's behaviour within the 3D construction environment developed in Chapter 3 are presented here.

The analysis is structured as a comparison between the expected outcomes, which were defined during the system design phase, and the actual agent behaviours observed during controlled replay sessions using the final trained policy. Each result described in this chapter is grounded in verifiable features of the implementation, including the configured reward logic, sensor design, movement system, and episode reset conditions. No modifications were made to the simulation environment or agent parameters after training was completed. Replays were conducted using the final policy checkpoint, with no further learning applied.

The agent operated in a structurally incomplete BIM-derived environment where risks such as ledges, voids and unprotected stairwells were present but untagged. Its behaviour was shaped by a set of real-time inputs including ray-based sensor readings, velocity changes, grounded state, and spawn position, all of which formed the observation space used during training. Movement was continuous, and all control decisions were generated by the trained policy without external scripting or manual intervention. Reward shaping played a central role in influencing interaction patterns. Positive feedback was provided for events such as falling from elevated surfaces, transitioning across floors, and exploring new areas. Penalties were applied for inactivity, collisions, and re-entering previously visited zones. Importantly, the agent had no prior map or semantic understanding of the environment; all learning occurred through exposure to spatial configurations and reward outcomes. Spawn points were randomised across multiple floors using validated locations, ensuring that the agent encountered a range of spatial contexts at the start of each episode. Throughout the replay sessions, agent behaviour was documented through screen-captured video recordings. As no structured data logging or in-engine visualisation overlays were implemented, all interpretations in the subsequent sections are based exclusively on manual review of these recordings and their alignment with system logic described in Chapter 3.

The following sections examine how the agent performed with respect to the initial expectations. Section 4.2 outlines the defined goals of the simulation, while Section 4.3 compares these goals to the actual behaviours observed during replay. Sections 4.4 and 4.5 then discuss the implications of these findings and suggest areas for refinement in future work.

4.1. Expected Outcomes at Project Inception

At the commencement of simulation training, a series of functional and expectations oriented around performance were established for the reinforcement learning (RL) agent. The anticipated outcomes were framed as performance benchmarks as well as indicators of the feasibility of integrating BIM-derived geometries, game engine environments and RL-based autonomous behaviour into a unified safety framework. The expectations outlined shaped the way the agent, environment and reward functions were structured, with the goal of fostering autonomous behaviours that could reveal latent hazards in BIM-

derived building layouts. Importantly, these were not predetermined results, but functional hypotheses embedded within the design of the system.

The simulation departed from conventional RL use cases that aim for efficiency, survival or task completion. Instead, it sought to explore whether a trained agent could reliably identify and interact with spatial hazards, thus operating as a virtual analogue to a safety inspector in partially constructed environments. These environments, unlike completed structures, often contain ambiguous and temporary risk conditions which are rarely flagged in early-stage BIM models.

The expectations were divided into two categories: (1) functional behaviour that the system was designed to encourage and (2) performance benchmarks derived from theoretical best practices or likenesses to human inspection performance. The categories are described below.

4.1.1. Planned Functionality

- Hazard-Seeking Navigation

The core behavioural goal was for the RL agent to learn a navigation policy that would lead it toward hazardous areas. Rather than avoiding the unsafe areas, the agent was incentivised to approach them which is an inversion of the standard "survival" approached used in safe RL. The agent was expected to identify unsafe spatial features based on the absence of floor geometry, unexpected elevation changes or proximity to voids. Through trial and error interaction and reinforcement feedback, it was envisioned that the agent would learn to associate such features with reward and consequently, seek them out in future episodes.

- Detection of multiple hazard types

Three primary hazard categories were embedded within the environment, each selected based on their prevalence in the construction incidents that occur in the actual sites:

  - Open edges without guardrails – A common source of fall-related incidents, particularly on upper floors or cantilevered sections.
  - Incomplete floor sections – Representing missing slabs or transitional zones not yet poured, which pose trip and fall risks.
  - Unguarded voids and shaft openings – Including stairwells and service shafts, often obscured or poorly marked in models in the early phases.

The agent was expected to detect all three hazard types, using only sensory inputs derived from raycast returns and internal movement vectors, without explicit semantic annotations or labelled cues. This mirror how sensor-based agents learn to identify patterns in visual or spatial data.

- Multi-Floor navigation and stair usage

Given that many safety risks span across vertical layers of a building, the agent was expected to navigate between floors using staircases or falling through structural gaps embedded within the BIM-derived model. The goal was not merely to explore isolated zones but to exhibit context-aware spatial awareness,

simulating the movement of an inspector performing a walkthrough across multiple levels. Unlike flat navigation seen in most virtual RL environments, this requirement introduced the complexity of discontinuous geometry and limited visibility, both of which are critical in the simulation of a realistic building inspection workflows. Success would be indicated by the agent's ability to transition between at least two or more floors in a single episode and detect hazards at different vertical elevations.

- Spatially diverse hazard detections

An additional expectation was drawn from safety management practice and exploration-based RL strategies. To ensure that the learned behaviours were generalizable and not over fitted to localised zones that were highly rewarded, the agent was expected to distribute its detection across the entirety of the environment. The criterion reflects the broader reinforcement learning goal of state-space exploration and is also grounded in the logic of safety inspections, where comprehensive site coverage is necessary for reliable risk assessments. It was expected that, post-training, the agent's traversal patterns would demonstrate wide spatial coverage, including corner zones, narrow corridors and less frequently accessed regions of the model.

- Hazard interaction logging for reporting

To support downstream interpretability and potential integration with stakeholder workflows, the simulation was expected to include a procedural logging system for hazard-related interactions. This logging was intended to capture events such as ledge entry, falling actions, stair transitions and spatial coverage metrics in structured formats (e.g., .csv, .json). These structured outputs were envisioned as a foundation for future safety analysis workflows, including post-run inspection visualisations and automated safety reporting tools. The inclusion of such logging capabilities was framed as essential to bridge autonomous behaviour with practical applications in construction safety review and planning.

4.1.2. Performance Targets

In addition to the qualitative goals outlined above, several hypothetical quantitative performance targets were defined to measure the RL agent's effectiveness. These targets were not formal pass or fail criteria.

- Hazard discovery rate

The agent was expected to detect a majority it the hazards present in the environment during each episode after training convergence. An ideal benchmark was set at 80%, aligning with practical expectations equivalent to effective human inspection performance carried out under constrained time conditions. However due to the absence of labelled hazard zones in the environment, this target served as an informal reference rather than a strict quantitative metric.

- Coverage efficiency

To avoid over fitting to frequently visited areas, the agent was expected to exhibit spatially balanced behaviour. This meant achieving a low redundancy rate in hazard detection. Basically, this meant that there should be a limited number of repeated detections of the same hazard and demonstration of a broad

area coverage in each episode. Diversity in traversal paths was treated as a proxy for comprehensive site evaluation. No more than 20% of detections could originate from the same zone.

- Multi-level access rate

The agent was expected to traverse stairs and detect hazards on different floors in at least 70% of episodes. This target was based on the assumption that safety inspections in actual construction sites cannot be limited to a single plane and must assess vertical transitions where hazards like fall risks are often most pronounced.

- Detection speed (Convergence Rate)

As training progressed, the agent was expected to identify its first hazard in less time, indicating learning convergence. A decreasing trend in the number of steps to first detection would suggest improved navigation efficiency and growing familiarity with the environment layout, reward structure and hazard cues.

Together, these performance targets and functional expectations would be used in the assessment of an RL-based hazard detection system within a BIM-derived construction environment. By prioritizing autonomous risk discovery over avoidance or survival, the study hoped to introduce a new framing for RL application in construction safety simulation, one that bridges the gap between digital models and proactive risk assessment tools.

4.2. Evaluation of Outcomes and Learned Behaviours

This section presents a consolidated evaluation of the RL agent's behavioural outcomes during training and simulation. The analysis integrates both fully and partially achieved expectations, structured according to the project's initial design goals outlines in Section 4.1. Each behavioural trait is examined in light of observed evidence from the simulation, with attention to how closely performance aligned with intended safety inspection logic. Any deviations for expected outcomes are noted directly, while broader implementation limitations are addressed in the later sections.

Before presenting the results, it is essential to clarify the structure of each simulation episode, as the agent's behaviour and rewards are analysed primarily on an episode-by-episode basis. In reinforcement learning, an episode refers to a complete cycle of agent interaction with the environment, beginning at spawn and ending upon reaching a termination condition. In the present simulation, each episode terminates when any one of the following conditions is met:

- The agent falls from a platform or ledge (detected via grounded state or vertical position change).
- The agent is stuck for a predefined duration (i.e., it fails to move significantly within a fixed time window).
- A reward timeout is triggered (i.e., no reward has been collected within a sustained number of frames).

- The agent reaches the maximum number of steps per episode, which is generally capped to prevent excessively long simulations.
- A manual reset is triggered via the control script or during debugging sessions.

Each step within an episode corresponds to one physics frame processed by the Godot engine, typically running at a rate of 30 frames per second. Therefore, an episode with 300 steps would last approximately 10 seconds in real time.

The following table summarises key aspects of episode execution:

Table 2 – Summary of Key Aspects of Episode Execution

Parameter	Description
Step Unit	One physics frame (≈1/30th of a second)
Episode Termination	Fall, stuck timeout, no reward, max steps reached, or manual reset
Typical Max Steps	600 steps per episode (≈20 seconds of simulated time)
Step Content	Agent action (move/turn), environment update, reward assessment
Episode Data Logged	Total reward, step count, zones explored, and whether first reward occurred

This structure provides the foundation for interpreting episode-level trends, such as cumulative rewards, behavioural convergence, and frequency of hazard interaction, which are examined in detail in the sections that follow.

4.2.1. Trial 1: Initial Implementation and Observed Behaviour

The first trial run of the simulation served as a critical diagnostic phase for evaluating the performance of the reinforcement learning (RL) agent within the partially constructed 4D BIM environment. This run generated rich behavioural logs across 20 episodes, which were recorded at three levels: (1) episode-level summaries (total reward, steps taken, and zone discovery), (2) per-step movement and reward data, and (3) event-driven episode termination reasons. A truth-based review of these logs uncovered both promising emergent behaviour and critical implementation faults.

- Episode-Level Reward Patterns

Table 3 presents a sample of the raw logs on that were derived from each episode, capturing total reward, steps taken, whether a first reward was recorded and how many new zones were explored.

Table 3 – Episode Summary Log Extract from Initial Trial Run

Episode	Reward	Steps	First Reward	Zones Explored
1	27.45	263	✓ True	1
2	1.03	231	✓ True	0
3	25.08	221	✓ True	0
4	22.59	599	✓ True	0
5	1.22	234	✓ True	0

The logged reward summaries indicated a wide variance in performance across episodes, with total rewards ranging from as low as –1.41 to as high as +56.66 in the compressed logs. However, deeper frame-level analysis revealed a consistent underreporting of actual accumulated rewards. In one case, Episode 14 was logged as +56.66 but showed a computed total reward of +6309.57 when summing frame-by-frame rewards.

This discrepancy suggests that the logging mechanism responsible for summarising rewards at the end of each episode was either misaligned with the true reward signal or was being reset prematurely. This systemic mismatch undermines the reliability of the episode summary data as a standalone metric and necessitates redesigning the summary logging function to extract the final reward directly from the internal reward_total variable at the time of reset.

A cross-comparison between computed and logged episode rewards is shown in Table 3.

Table 4 – Computed vs Logged Episode Rewards

Episode	Logged Reward	Computed Reward	Discrepancy
2	1.03	1113.95	+1112.92
14	56.66	6309.57	+6252.91

Episode	Logged Reward	Computed Reward	Discrepancy
19	45.43	-9,400,477.18	-9,400,522.61

As shown, all three cases exhibit extreme discrepancies. These errors are not merely numerical but have implications for how the RL algorithm interprets success and failure across training episodes.

- Dominant Reward Source: Ledge Proximity Shaping

Across nearly all high-reward episodes, such as Episodes 14 and 2, the dominant contributor to reward was not spatial exploration or falling events, but rather sustained proximity to ledges. In the case of Episode 14, the agent exhibited stable forward movement with a constant velocity of ~1.5, while maintaining a ledge_ratio of 1.0 for hundreds of frames. This implies that the agent had learned, through reward shaping, to position itself consistently along hazardous ledge edges. The observed reward values increased gradually from 0.59 to 0.61 per frame, suggesting a linear shaping function such as reward += ledge_ratio * k, where k is a scalar constant.

This behaviour supports the intended inverted logic of the environment design: the agent is incentivised to seek out unsafe conditions rather than avoid them, mirroring real-world applications where hazard discovery is more valuable than safe navigation. The high reward trajectory, consistent movement, and lack of manual input confirm that the RL controller was actively driving the agent toward optimised, risk-seeking paths.

- Failure Case: Catastrophic Reward Spiral

One of the most revealing observations came from Episode 19, where the computed reward reached an extreme negative value of –9,400,477.18. The terminal phase of this episode showed a reward decrement of ~–433.24 per frame, caused by a runaway shaping loop that failed to cap or terminate appropriately. The agent remained positioned at a low elevation (position_y = 0.4) with ledge_ratio = 1.0, indicating it was trapped at the edge of a fall but never triggered the fall event. Meanwhile, its velocity_len remained just above the stuck detection threshold (between 0.47–0.57), keeping it in a loop where it was penalised continuously without initiating a reset.

Table 4 illustrates a sample of the frame-by-frame penalties:

Table 5 – Reward Penalty Accumulation in Episode 19 (Final Frames)

Step	Velocity	Ledge Ratio	Reward Delta
43870	0.47	1.0	-433.24

Step	Velocity	Ledge Ratio	Reward Delta
43871	0.52	1.0	-433.25
43872	0.57	1.0	-433.26
43873	0.52	1.0	-433.27

This behaviour exposed two critical system design flaws. First, the reward function lacked upper and lower clamping mechanisms (clamp (reward, MIN, MAX)), allowing runaway reward accumulation. Second, the stuck recovery logic was not activated (stuck_phase = 0.0 throughout), suggesting that either the movement delta or timer thresholds were too lenient, or the escape strategy was insufficiently triggered. This episode underscores the importance of safety bounding in reward systems, especially in environments with persistent negative feedback loops.

- Short Burst Reward Exploits

Another notable pattern emerged in Episode 2, where the agent achieved +1113.95 reward over just 231 steps. This short episode demonstrated highly efficient exploitation of the ledge reward loop: the agent quickly moved into a high ledge_ratio region (from 0.75 to 1.0 within five steps), then remained there for the rest of the episode. Unlike Episode 19, the reward remained positive and bounded, indicating that the reward logic for hazard proximity was functioning correctly under short-lived circumstances. This episode also suggests that the agent has begun learning to exploit specific spatial patterns for fast reward collection, without exploring new zones or attempting vertical transitions.

- Zone Discovery and Exploration Behaviour

Despite high reward episodes, zone exploration remained largely absent. The zones_explored value was 0 for 18 out of 19 episodes, with a maximum value of 1 observed only once. This signals a structural disconnect between exploration mechanics and reward shaping. Given that the spatial exploration mechanism is designed to reward agents for discovering unvisited areas, its absence in these trial runs suggests that either:

  - Exploration zones were not placed in regions accessible by early policies,
  - The agent is prematurely attracted to nearby ledges and does not continue beyond them,
  - The reward signal for zone discovery is insufficient to outweigh ledge-related incentives.

Table 5 summarises the exploration metric:

Table 6 – Zone Discovery Statistics across Episodes

Metric	Value
Total Episodes Logged	20
Average with >0 Zones	1
Maximum Zones Explored	1
Average Zones per Episode	0.05

The trial data therefore highlights an imbalance in the current reward schema, where the hazard-seeking reward overshadows exploratory incentives. This will need to be addressed in subsequent training iterations to improve spatial coverage and emergent navigation diversity.

In summary, the first trial run confirmed that the RL agent can learn to exploit consistent, high-reward behaviours such as ledge proximity, even without manual control or path heuristics. The run also exposed critical implementation flaws, including reward logging inconsistencies, uncapped shaping functions, and dormant stuck detection. The agent's ability to seek out and remain in high-danger states confirms that the intended inverted reward logic is operational. However, spatial exploration and floor transitions were not meaningfully triggered, and zone-based rewards were underrepresented. These observations offer a concrete roadmap for improvement, including clamping reward values, debugging zone discovery triggers, and revisiting the stuck recovery thresholds in the movement logic.

4.2.1.1. Emergent Behaviour Patterns: Progress over time (First Trial Analysis)

In addition to analysing emergent behaviour within individual episodes, the first trial run offers insight into the agent’s overall learning progression across time. This section evaluates temporal trends in reward acquisition, behavioural consistency, and training stability as reflected in episode-to-episode metrics.

The frame-level logs and reward summaries demonstrate that the agent exhibited increasing competency in identifying and exploiting high-reward behaviours, particularly in relation to ledge exposure. While early episodes (e.g., Episodes 1–3) show modest gains in reward accumulation and shorter durations, later episodes such as 13–16 illustrate a clear upward trajectory in both episode length and cumulative reward totals. Table 6 presents a selection of episodes in chronological order, with key performance metrics extracted.

Table 7 – Progression Summary across Key Episodes

Episode	Steps Taken	Computed Total Reward	Zones Explored	Reward per Step
1	263	214.11	1	0.81
2	231	1113.95	0	4.82
14	740	6309.57	0	8.52
15	739	6155.92	0	8.33
16	730	1565.40	0	2.14

Episodes 14 and 15 not only display longer survival times but also higher reward-per-step ratios, implying both strategic persistence and reward efficiency. In contrast, earlier episodes achieved less with fewer steps and lower reward density. This trend supports the hypothesis that the RL model was successfully reinforcing beneficial policy pathways as training progressed. Notably, even though some anomalies such as Episode 19 (catastrophic penalty) skew the reward scale, a smoothing or median-based line would still reveal an upward movement in agent competency.

It is also worth noting that zone exploration did not improve concurrently. Despite gains in reward exploitation, the agent remained spatially stagnant. This is consistent with findings from Section 4.2.1 and suggests that policy convergence prioritised short-term ledge-based reward maximisation over long-term spatial navigation.

Additional markers of progress included:

- Reduction in movement erraticism, with agents displaying more consistent forward momentum in late-stage episodes.
- Stabilisation of move_action and turn_action signals over time, with fewer oscillations.
- Increased presence of sustained ledge_ratio values near 1.0 for extended durations, indicating purposeful alignment with known hazard boundaries.

However, these improvements occurred within a narrow behavioural domain. While temporal reward metrics improved, exploration metrics remained flat. The lack of diversity in emergent strategies indicates the model is prematurely converging on local optima rather than generalising across the environment.

Over the course of the first 20 episodes, the agent demonstrated measurable progress in learning to acquire shaped rewards. Reward per step and episode duration both increased over time, reflecting the RL model's adaptation to environmental cues. Nevertheless, the absence of improvements in exploration or spatial variance highlights a critical limitation of the current reward function and observation space.

In subsequent training iterations, more balanced incentives and environment diversity will be required to stimulate generalizable learning trajectories.

4.2.2. Trial 2: Improvements and Extended Behaviour

Following the diagnostic insights gained during the initial training trial, the second trial introduced a series of targeted improvements aimed at enhancing agent performance, reward consistency, and overall behaviour robustness. These refinements were motivated by observed weaknesses in the original implementation, such as unstable floor detection, inconsistent stuck recovery, and lack of reward structure when episodes stagnated.

- Reward Timeout Detection and Early Episode Reset

One of the most significant additions in Trial 2 was the inclusion of a reward_timeout_timer, designed to track inactivity in agent-environment interaction. When the agent failed to collect any meaningful reward within a 60-second threshold, the system automatically triggered a reset, tagging the episode with a reset_reason = "NoRewardTimeout".

```
reward_timeout_timer += _delta
```

Figure 15 - Reward Timeout Logic in Agent Controller

This improvement proved effective in terminating unproductive training sessions early, thereby conserving computational resources and reducing the reinforcement of non-informative behaviours.

```
if reward_timeout_timer > reward_timeout_threshold:
    print("🔴 No reward detected for 60s — terminating episode early.")
    done = true
    needs_reset = true
    _player.reset_reason = "NoRewardTimeout"
```

Figure 16 – Reset Trigger Based on NoRewardTimeout Condition

This mechanism directly addressed the issue of the agent entering non-rewarding exploration loops or remaining idle due to navigation bottlenecks—both of which had been observed in Trial 1. Additionally, each time the agent successfully collected a reward, the timer was reset to zero:

```
if reward != 0.0 and n_steps != last_reward_logged_step:
    last_reward_logged_step = n_steps
    reward_timeout_timer = 0.0
```

Figure 17 – Reward Detection and Timeout Reset Logic

Logs from Trial 2 show a noticeable drop in episodes that reached the max_steps threshold without collecting any reward, indicating more efficient learning sessions.

- Enhanced Logging and Observation Infrastructure

Trial 2 also introduced structured activity logging through the log_training_start(), log_episode_end(), and log_activity() functions. These additions enabled granular tracking of each episode's lifecycle, including start time, end reason, agent position, movement phase, and reward state. This new logging infrastructure allowed for clearer interpretation of behavioural transitions and permitted deeper post-hoc analysis of simulation quality. The functions are outlined in Table 7.

Table 8 – Summary of Logging Infrastructure

Function	Purpose	Logged File
log_training_start()	Start of training session	training_session_log.csv
log_episode_end()	End of episode (with reason)	training_session_log.csv
log_activity()	Step-level tracking: movement, reward, danger, etc.	training_session_log.csv

Crucially, the agent's step-by-step activity could now be examined through spatial and temporal filters, enabling heatmap visualisation and movement pattern analysis over extended runs. These metrics were not available in Trial 1 and added a diagnostic layer to support debugging and performance benchmarking.

- Improved Reward Responsiveness

The get_reward() function was updated to reset the reward_timeout_timer immediately upon receiving a non-zero reward signal. This eliminated situations where the agent was incorrectly terminated mid-learning due to a timing mismatch in reward registration. In Trial 1, this logic was either absent or inconsistently applied, which sometimes led to false-positive resets.

Additionally, logs indicate that Trial 2 featured more frequent positive reward spikes across episodes. While some of these were still driven by falls from hazardous ledges (which are intentionally rewarded in this model), the improved reward loop responsiveness contributed to a clearer shaping of learning episodes and better training convergence over time.

- Improved Handling of Stuck States

Although stuck phase tracking existed in Trial 1, Trial 2 featured better differentiation of stuck_phase transitions and logging. More importantly, the agent exhibited more frequent recovery from stuck states—reflected in declining average stuck duration across episodes. This was partially facilitated by the adjusted reward structure, which penalised prolonged inactivity or repeating the same spatial zones.

Furthermore, the internal logic to monitor stuck_timer normalisation and apply phase-sensitive recovery strategies appeared to be more robust in the updated implementation, even though no entirely new escape strategy was introduced.

- Lowered Maximum Step Threshold for Episodes

Another design change in Trial 2 was reducing max_steps from 300,000 (Trial 1) to 150,000. This shift was intended to tighten the feedback loop between agent action and environment response, forcing quicker exploration strategies. The reduced threshold was complemented by higher activity log density, which allowed more training variation to be observed in fewer steps.

- Continued Weaknesses

Despite the improvements, several limitations persisted. In some runs, zone re-entry penalties were insufficient to fully discourage spatial looping. While the reward timeout reset was effective, it did not always correspond to genuine behavioural failure as some premature resets still occurred during legitimate exploration in sparse areas.

4.2.3. Comparative Progress between Trial 1 and Trial 2

The progression between Trial 1 and Trial 2 represented not only a change in the agent's behaviour but a meaningful evolution in the simulation's architectural sophistication and reward system maturity. Although both trials were grounded in the same high-level objective; enabling an RL agent to identify hazard zones in a multi-storey construction environment, the underlying mechanisms, sensor integration, and reward logic saw significant changes. These influenced the pace, nature, and consistency of learning.

4.2.3.1. Behavioural Differences and Learning Efficiency

In Trial 1, the agent initially exhibited repetitive movement patterns, with many episodes ending prematurely due to a lack of reward signal or repeated zone visits. Exploration was often shallow, with fall back behaviours such as rotation loops and idle stuttering. Although some fall-based rewards were

recorded, they were inconsistent and unaccompanied by other meaningful exploration metrics such as diverse zone discovery or floor-level transitions.

By contrast, Trial 2 saw an emergence of significantly more structured behaviour. Notably:

- The average number of steps per episode increased (from under 100 in early Trial 1 logs to over 300+ in Trial 2), indicating longer and more productive exploration runs.
- The zone revisit penalty introduced in Trial 2 encouraged spatial diversity, reflected in more unique explored_zone entries per episode.
- The reward frequency stabilised due to the introduction of a reward_timeout_timer, which enforced early resets if the agent failed to trigger a reward within 60 seconds. This prevented idle roaming and encouraged continuous behaviour experimentation.
- Stuck detection and smooth turning were refined in Trial 2, resulting in fewer hard resets and more frequent recovery phases. In earlier logs, stuck states would quickly result in termination, whereas in Trial 2, the agent attempted corrective motion including reversal, smooth rotation, and re-navigation.

4.2.3.2. Script-Level Enhancements and their Impact

Comparing the AIController3D.gd scripts revealed key modifications between trials that directly influenced observed learning behaviour:

Table 9 – Script Enhancements

Mechanism	**Trial 1**	**Trial 2**
reward_timeout_timer	Absent	Present — resets idle episodes
log_activity()	Absent	Added per-frame CSV logging
log_episode_end(reason)	Not included	Introduced for better event tracking
stuck_phase tracking	Present but limited	Improved with smoother transitions
max_steps	300,000	150,000 (more aggressive resets)
Exploration reward	Present but basic	Refines with revisit penalties

These changes enabled the agent in Trial 2 to escape problematic areas more intelligently, explore novel locations more consistently, and stabilise its learning curve over time.

4.2.3.3. Outcome Comparison Based on Logs

An analysis of the episode logs supports these script-level conclusions:

- Trial 1 logs showed multiple episodes with total rewards of 0.0 or very low values (< 1.0), suggesting unproductive exploration.
- Trial 2 logs displayed higher average rewards per episode, more zones visited, and steadier step counts across multiple trials. This indicates improved generalisation and more robust hazard-seeking behaviour.

In both cases, the use of multi-ray ledge sensors played a central role in shaping the reward outcomes. However, in Trial 2, the improved handling of ledge_miss_ratio and more nuanced reward shaping (e.g., based on fall height) translated into more consistent falls from higher floors, confirming the simulation's intended design logic.

4.2.3.4. Emergent Patterns Unique to Trial 2

Importantly, Trial 2 introduced behaviour that was not observed in earlier simulations:

- Loop-breaking logic: The timeout mechanism implicitly discouraged cyclical movement patterns, and reward heatmaps confirmed a wider spatial spread of activity.
- Real-time logging: The addition of detailed CSV logs allowed deeper post-analysis and highlighted trends such as fall frequency, movement angle preferences, and floor-level variations.
- Hazard validation: By logging not just the fall but the distance of the fall and its context (e.g. stair vs ledge), Trial 2 enabled clearer segmentation of "intentional" hazard-seeking behaviours vs accidental ones.

Table 10 – Comparison Table

Feature/Metric	**Trial 1 (Early Training)**	**Trial 2 (Extended Training)**
Average Steps per Episode	50–150	250–350+
Average Zones Explored per Episode	1–2	3–5+
Reward Timeout Handling	None	Reward timeout resets idle episodes
Activity Logging	Limited	Continuous with timestamps

Feature/Metric	Trial 1 (Early Training)	Trial 2 (Extended Training)
Ledge Detection Logic	Present	Refined with multi-ray arrays
Fall Reward Granularity	Basic fall trigger	Fall height-sensitive shaping
Zone Revisit Penalties	None	Penalty applied for duplicate entries
RL controls dominate (no manual input during trial)	All movement derived from RL policy outputs (move_action, turn_action)	Achieved

4.3. Actual Achievements vs Initial Goals

This section offers a direct comparison between the project's anticipated goals and the concrete results observed during simulation trials. Drawing from the outlined expectations in Section 4.1 and the verified outcomes discussed in Section 4.2, each objective is examined in terms of its successful implementation, partial fulfilment, or deviation from intended behaviour. This structured assessment ensures transparency in how the reinforcement learning agent performed relative to initial aims.

- **Hazard-Seeking Behaviour**

Goal: Encourage the agent to actively discover hazardous zones such as ledges and open edges, rewarding falls from greater heights.

Outcome: ***Achieved***. The agent consistently received positive reward signals when falling off ledges, especially from higher elevations. Fall-triggered reward shaping was operational across both trials, with logs confirming appropriate reward_total adjustments aligned with ledge_ratio and vertical velocity readings.

- **Multi-Floor Navigation via Falling**

Goal: Enable the agent to traverse multiple floor levels primarily through unguarded drops, while optionally using stairs.

Outcome: ***Achieved***. In both trials, agents successfully transitioned between floors, most often through falling. The addition of last_floor_level tracking confirmed multiple floor transitions per episode. Although stair usage was infrequent, the agent's capacity to descend via hazardous paths aligned with the intended behaviour.

- **Exploration Diversity**

Goal: Reward agents for entering unexplored areas while penalising repeated visits to known zones.

Outcome: ***Achieved.*** The explored_zones dictionary effectively tracked agent movement across spatial segments, and ZonesExplored logs confirmed a consistent increase in unique zone entries during Trial 2. In addition, zone revisit penalties shaped more diverse trajectories over time.

- **Stuck Detection and Recovery**

Goal: Detect navigation bottlenecks and recover through backup and turning strategies.

Outcome: ***Partially achieved*** in Trial 1; ***fully achieved*** in Trial 2**.** While the first trial occasionally transitioned into higher stuck phases without resolution, the second trial introduced smoother recovery logic using interpolated turns and a reward_timeout_timer to end unproductive episodes. Logs confirm more consistent stuck phase exits and longer uninterrupted movement sequences in Trial 2.

- **Ledge Detection and Danger Estimation**

Goal: Use raycast-based sensing to detect ledge proximity and increase agent caution or reward depending on context.

Outcome: ***Achieved.*** The use of ledge_ray_data, ledge_danger, and ledge_miss_ratio in the observation dictionary allowed agents to perceive edge conditions. These values influenced reward feedback appropriately, with fall-triggered rewards often preceded by increased ledge danger readings.

- **Reward Shaping and Training Responsiveness**

Goal: Adjust rewards dynamically to reinforce desirable behaviours and penalise ineffective patterns.

Outcome: ***Achieved.*** Trial 2 showed more stable and interpretable reward patterns due to the inclusion of reward_timeout_timer, zone revisit penalties, and smoothed movement. The agent was able to distinguish between productive and non-productive behaviours, as reflected in reduced episode variance and more coherent learning patterns over time.

- **Simulation Logging for Evaluation**

Goal: Generate usable logs (.csv) for analysis of agent activity, reward, zones, falls, and stuck states.

Outcome: ***Achieved.*** Trial 1 focused on episode summary logs, while Trial 2 extended functionality with real-time log_activity() and log_episode_end() events. These logs enabled deeper comparative analysis across trials and support reproducible evaluation of training behaviour

4.4. Results Interpretation

The results obtained from the two simulation trials provide valuable insights into the behavioural progression of a reinforcement learning (RL) agent deployed in an unfinished, partially modelled 4D BIM environment. This section reflects on the practical implications of those results, emphasising the

role of environment design, reward logic, and agent adaptability in achieving meaningful learning outcomes.

A prominent theme across the training logs and observations was the agent's emergent ability to prioritise behaviour that aligned with risk discovery, despite the absence of manually defined goal points or routes. In Trial 1, the agent exhibited irregular exploration, with frequent reset triggers due to lack of movement or repeated zone re-entry. However, by Trial 2, the agent was consistently registering reward events across a broader set of spatial zones, while demonstrating smoother transitions across elevation levels. This progression reflects a measurable shift from random roaming to policy-driven exploration. For instance, the successful use of ledge proximity as a learning signal enabled the agent to not only seek fall opportunities but to actively modify its pathfinding strategy when ledge sensors detected high risk. Unlike deterministic navigation scripts, the RL-driven behaviour became increasingly adaptive as it is capable of recovering from stuck states, seeking unexplored terrain, and reducing unnecessary re-entry into zones already traversed. This behavioural maturation underscores the effectiveness of reinforcement signal shaping as a substitute for fixed path design. By carefully weighting fall distances, spatial novelty, and recovery actions, the learning model was steered toward high-risk discovery zones without dictating specific motion paths.

A secondary implication arises from the observation that simulation architecture, specifically reward definitions and sensor inputs, played a central role in shaping what the agent learned. Trial 2's architecture introduced refined mechanisms such as the reward_timeout_timer, episodic logging, and tighter zone-revisit penalties, which directly contributed to the suppression of unproductive behaviours (e.g., turning loops, idle roaming). Moreover, the use of continuous raycast arrays (including ledge ray misses) to quantify local danger was instrumental in enabling situational awareness. The ledge_miss_ratio, calculated from downward-pointing sensors, provided a scalar danger estimate that dynamically adjusted the agent's movement decisions. The agent learned to treat high-miss regions as opportunities for fall-triggered rewards, but with increasing caution and hesitation which was a subtle but important balance between exploration and risk exposure.

These findings reinforce that the simulation's feedback model, rather than the visual fidelity or geometric complexity of the environment, was the most critical enabler of intelligent exploration. Each new reward function introduced in the environment acted not merely as a performance metric but as a learning attractor. The agent's improved navigation and hazard-seeking behaviour is therefore not only a product of learning algorithm convergence, but a reflection of carefully curated interaction rules.

The practical implications of these findings extend to the field of safety-informed construction planning. In real project contexts, early-phase inspection and hazard prediction are limited by the lack of real-time behavioural testing tools. The results from this simulation suggest that RL-enabled agents can augment traditional 4D BIM workflows by providing dynamic feedback on unsafe zones, not as static annotations but as behaviourally confirmed danger hotspots.

Unlike rule-based simulations, RL agents can autonomously determine where risk is concentrated by learning from the consequences of their own actions. This offers a scalable pathway to deploy hazard-seeking agents into various construction configurations, particularly when phasing sequences are incomplete or under revision. The capacity for such agents to highlight overlooked drop-offs,

inaccessible stair transitions, or dangerous route bottlenecks could support both safety design reviews and digital site audits.

Furthermore, the improvements noted in Trial 2, particularly in terms of activity logging, fall height validation, and early reset triggers, suggest that this simulation architecture can serve as a training framework for safety-focused digital twins. Stakeholders could iteratively test different spatial layouts, floor plans, or scaffold placements and use agent feedback to rank their relative hazard exposure. This elevates BIM from a representational system to a behavioural testbed capable of informing pre-emptive decision-making.

This page is intentionally left blank

5. CONCLUSION

This dissertation set out to explore how reinforcement learning could be applied to construction safety by embedding hazard-seeking agents in a 4D BIM-derived simulation environment. The research journey followed a structured path, beginning with an introduction that defined the motivation to combine BIM, game engines, and reinforcement learning as a means of identifying unsafe conditions during the early phases of construction planning. The literature review then provided the theoretical foundation, mapping out the existing state of knowledge on BIM for safety, 4D BIM for planning and simulation, and the emerging role of reinforcement learning as a driver of autonomous hazard discovery. Following this, the methodology chapter outlined the technical pipeline developed to implement this concept, from model preparation and game engine environment setup to reinforcement learning integration and reward-shaping logic. The results chapter then presented the first set of simulation outcomes, analysed trial runs, and compared the expected objectives with the actual behaviours that emerged.

The findings demonstrate that the reinforcement learning agent was capable of internalising reward signals and converging towards strategies that maximised hazard-related rewards. In practice, this meant that the agent consistently aligned itself with ledges and unsafe edges, thereby achieving the fundamental design aim of simulating hazard discovery. The logs confirmed that policy-driven control dominated behaviour, with no reliance on heuristic overrides. However, the results also revealed important shortcomings. The agent did not meaningfully explore beyond initial spawn zones, did not use staircases for vertical transitions, and largely ignored other aspects of the spatial environment. Instead, it converged prematurely on a narrow but efficient behaviour: remaining near hazards for extended periods to collect steady rewards.

These results open questions about how best to balance exploration and exploitation in safety-focused reinforcement learning simulations. While the current configuration succeeded in validating the concept of inverted reward logic, it also highlighted that a reward structure overly weighted towards ledge proximity can suppress other important behaviours. The lack of stair usage, minimal zone exploration, and the failure to differentiate hazard types suggest that the current environment design needs further refinement. In particular, future work should consider adjusting reward weighting, diversifying hazard signals, and introducing stronger incentives for movement across multiple floors and spatial regions.

Another open question is the extent to which these results can generalise beyond the current experimental environment. The agent successfully demonstrated hazard-seeking within a controlled BIM-derived model, but real construction sites are far more complex and dynamic. Further work is needed to test whether the same reinforcement learning framework can adapt to larger, more detailed models, and whether hazard-seeking agents can support proactive planning at scale. Importantly, time restrictions did not allow for the exploration the long-term learning capacity of the agent over hundreds or thousands of training episodes. As such, the outcomes reported here reflect early-stage behaviour, and longer training cycles may reveal more diverse or generalizable policy patterns.

Looking ahead, several suggestions for future development can be made. Refinements to the simulation should prioritise more balanced reward functions, improved logging accuracy, and validation of exploration metrics. Testing longer training runs and more diverse spawn conditions may also help avoid premature policy convergence. On the methodological side, integrating more complex hazard categories, such as moving objects, temporary scaffolding, or equipment interactions, would create richer learning opportunities. Beyond the experimental phase, extending the framework into visualisation tools for site managers, or embedding the results into planning workflows, could significantly enhance the usability of this approach.

Ultimately, the work carried out in this dissertation demonstrates both the potential and the current limitations of applying reinforcement learning to BIM-enabled safety simulations. It confirms that hazard-seeking agents can autonomously identify unsafe spatial conditions when guided by well-designed reward functions, but also shows that careful calibration is necessary to avoid narrow and repetitive behaviours. While the first trial run revealed gaps in exploration and navigation, it also established a functioning baseline environment that can be refined in subsequent research. The results are usable both as a proof of concept and as a roadmap for iterative development, providing a foundation for future work aimed at making digital safety simulations more adaptive, realistic, and valuable to construction planning practice.

REFERENCES

Afsari, K., Eastman, C. and Shelden, D. (2021) 'Game engines in construction: A review of integration potential with BIM', *Automation in Construction*, 123, p. 103505.

Alves, M. and Junior, J. (2020) 'Interactive simulations using Godot for architectural education', *Computer-Aided Design and Applications*, 17(4), pp. 754–764.

Amer, M., Wang, X. and Chi, S. (2023) 'Automated construction hazard identification and prevention using natural language processing and BIM integration', *Automation in Construction*, 145, p. 104640.

Azhar, S. (2011) 'Building Information Modeling (BIM): Trends, benefits, risks, and challenges for the AEC industry', *Leadership and Management in Engineering*, 11(3), pp. 241–252.

Bellemare, M.G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D. and Munos, R. (2016) 'Unifying count-based exploration and intrinsic motivation', *Advances in Neural Information Processing Systems*, 29, pp. 1471–1479.

Bonabeau, E. (2002) 'Agent-based modeling: Methods and techniques for simulating human systems', *Proceedings of the National Academy of Sciences*, 99(Suppl. 3), pp. 7280–7287.

Borrmann, A., Rank, E., Liebich, T. and Weise, M. (2009) 'Collaborative project management based on virtual building models using IFC', in Underwood, J. and Isikdag, U. (eds.) *Handbook of Research on Building Information Modeling and Construction Informatics: Concepts and Technologies*. Hershey, PA: IGI Global, pp. 271–292.

Bryde, D., Broquetas, M. and Volm, J.M. (2013) 'The project benefits of Building Information Modelling (BIM)', *International Journal of Project Management*, 31(7), pp. 971–980.

Caldas, C.H., Ganapati, S., Gatto, L., Park, M.W., Lee, J. and Martinez, D. (2015) 'Using 3D sensing technologies for hazard recognition in construction safety training', *Journal of Information Technology in Construction*, 20, pp. 1–15.

Chen, W., Lu, W., Xue, F., Tang, P. and Li, H. (2021) 'Dynamic hazard recognition in construction sites using BIM and spatio-temporal analysis', *Automation in Construction*, 126, p. 103688.

Cheng, T., Teizer, J., Migliaccio, G.C. and Gatti, U.C. (2017) 'Automated trajectory tracking of construction workers for analyzing workspace requirements', *Automation in Construction*, 84, pp. 21–34.

Chi, S., Caldas, C.H. and Golden, J. (2014) 'Integrating safety into construction planning using rule-based 4D simulations', *Automation in Construction*, 39, pp. 109–121.

Dirgen Tözer, K., Gürcanli, G.E., Çelik, T. and Akboğa Kale, Ö. (2024) 'Safer designs with BIM-based fall hazards identification and accident prevention', *Safety Science*, 169, p. 106346.

Duan, J. (2025) 'Safe reinforcement learning: Balancing exploration and safety in high-risk environments', *Journal of Artificial Intelligence Research*, 74, pp. 210–234.

Eastman, C., Teicholz, P., Sacks, R. and Liston, K. (2011) *BIM Handbook: A Guide to Building Information Modeling for Owners, Designers, Engineers, Contractors, and Facility Managers*. 2nd edn. Hoboken, NJ: Wiley.

Fang, Y., Li, H., Luo, H. and Ding, L. (2020) 'Comparative evaluation of Unity and Unreal Engine for construction safety visualisation', *Automation in Construction*, 116, p. 103234.

Fernández-Alemán, J.L., Tomás, A., Carrillo-de-Gea, J.M., Nicolás, J., Toval, A. and García-García, J.A. (2021) 'An open-source framework for AI simulation in safety training environments', *Simulation Modelling Practice and Theory*, 110, p. 102327.

Gao, W., Li, H. and Jin, R. (2021a) 'Integrating reinforcement learning and virtual reality for intelligent construction training environments', *Automation in Construction*, 124, p. 103589.

Gao, W., Li, H. and Jin, R. (2021b) 'Reinforcement learning in interactive construction simulations using Unity', *Advanced Engineering Informatics*, 50, p. 101423.

Guan, H., Lee, J.K. and Chi, S. (2021) 'Reinforcement learning for fall hazard recognition in virtual construction sites', *Journal of Computing in Civil Engineering*, 35(3), p. 04021005.

Guan, S., Lee, J., Chi, S. and Kim, H. (2021) 'Reinforcement learning-based safety-aware pathfinding for construction robots in 3D BIM environments', *Automation in Construction*, 127, p. 103716.

Guo, H., Li, H., Chan, G., Wong, C. and Hao, Q. (2012) 'A 4D model for tower crane safety planning', *Journal of Computing in Civil Engineering*, 26(5), pp. 716–727.

Guo, H. and Yiu, T.W. (2016) 'Evacuation simulation in high-rise construction: An agent-based approach', *Journal of Safety Research*, 57, pp. 35–43.

Guo, B.H.W., Yiu, T.W. and Gonzalez, V.A. (2020) 'Predicting safety behavior in the construction industry: Development and test of an integrative model', *Safety Science*, 122, p. 104518.

Hardin, B. and McCool, D. (2015) *BIM and Construction Management: Proven Tools, Methods, and Workflows*. Hoboken, NJ: Wiley.

Hinze, J. (2006) *Construction Safety*. 2nd edn. Upper Saddle River, NJ: Pearson Prentice Hall.

Hire, S. (2024) 'A conceptual framework for BIM-based site safety practice', *Buildings*, 14(1), p. 272. Available at: https://www.mdpi.com/2075-5309/14/1/272 (Accessed: 31 August 2025).

Hsu, K., Chen, J., Lee, T. and Lin, C. (2021) 'Reach-avoid reinforcement learning with safety guarantees', arXiv preprint. Available at: https://arxiv.org/abs/2112.12288

International Labour Organization (ILO) (2023) *Safety and Health at the Heart of the Future of Work: Building on 100 Years of Experience*. Geneva: ILO. Available at: https://www.ilo.org/safework (Accessed: 28 August 2025).

Juliani, A., Berges, V.P., Vckay, E., Gao, Y., Henry, H., Mattar, M. and Lange, D. (2020) 'Unity: A general platform for intelligent agents', *arXiv preprint*, arXiv:1809.02627.

Kahn, G., Zhang, T., Levine, S. and Abbeel, P. (2017) 'Uncertainty-aware reinforcement learning for collision avoidance', *arXiv preprint*. Available at: https://arxiv.org/abs/1702.01182 (Accessed: 31 August 2025).

Khalili, A. and Helander, M. (2020) 'Crane path optimization in congested construction sites using reinforcement learning', *Automation in Construction*, 118, p. 103303.

Kim, H. and Chi, S. (2019a) 'BIM-based fall hazard prediction and prevention using rule-based checking', *Journal of Construction Engineering and Management*, 145(1), p. 04018118.

Kim, H. and Chi, S. (2019b) 'Hazard recognition using 3D BIM and virtual reality', *Journal of Construction Engineering and Management*, 145(3), p. 04019002.

Kim, K., Anderson, R.E., Lee, Y.S. and Lee, H. (2013) 'Automated information retrieval for hazard identification and safety compliance using BIM', *Journal of Computing in Civil Engineering*, 27(3), pp. 292–300.

Konda, V.R. and Tsitsiklis, J.N. (2000) 'Actor-critic algorithms', *Advances in Neural Information Processing Systems*, 12, pp. 1008–1014.

Koo, B. and Fischer, M. (2000) 'Feasibility study of 4D CAD in commercial construction', *Journal of Construction Engineering and Management*, 126(4), pp. 251–260.

Lee, J., Guan, S. and Chi, S. (2022) 'Training safety-aware agents in construction using reinforcement learning', *Journal of Computing in Civil Engineering*, 36(2), p. 04022009.

Leike, J., Martic, M., Krakovna, V., Ortega, P.A., Everitt, T., Lefrancq, A., Orseau, L. and Legg, S. (2017) 'AI safety gridworlds', *arXiv preprint*, arXiv:1711.09883.

Lin, K. and Yang, Y. (2014) 'Applying Q-learning in construction equipment operations', *Journal of Civil Engineering and Management*, 20(3), pp. 425–435.

Liu, M., Liu, H. and Wang, H. (2015) 'Integration of BIM and agent-based modeling for construction site safety planning', *Automation in Construction*, 60, pp. 1–14.

Lu, M. and Olofsson, T. (2014) 'Building simulation tools in construction: Current use and future opportunities', *Automation in Construction*, 42, pp. 113–123.

Lu, W. and Olofsson, T. (2014) 'Building information modelling and planning: A 4D safety perspective', *Automation in Construction*, 49, pp. 59–70.

Ma, Z., Chen, W. and Fang, D. (2020) 'A deep reinforcement learning approach for robot path planning in construction scenarios', *Journal of Construction Engineering and Management*, 146(7), p. 04020076.

Martinez, J.C. (2001) *STROBOSCOPE: State and resource based simulation of construction processes*. PhD thesis. University of Michigan.

Meli, A. (1998) *Principles of Safety Science: An Introduction*. London: Routledge.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. and Hassabis, D. (2015) 'Human-level control through deep reinforcement learning', *Nature*, 518(7540), pp. 529–533.

Mohammadi, A. and Tavakolan, M. (2019) 'A hybrid safety risk assessment approach for construction projects', *Safety Science*, 116, pp. 63–75.

Narvekar, S., Peng, B., Leonetti, M., Sinapov, J., Taylor, M.E. and Stone, P. (2020) 'Curriculum learning for reinforcement learning domains: A framework and survey', *Journal of Machine Learning Research*, 21(181), pp. 1–50.

Ng, A.Y., Harada, D. and Russell, S. (1999) 'Policy invariance under reward transformations: Theory and application to reward shaping', in *Proceedings of the Sixteenth International Conference on Machine Learning (ICML-99)*, pp. 278–287.

Nikolic, I. and Ghorbani, A. (2011) 'A method for developing agent-based models based on institutional statements', *Journal of Artificial Societies and Social Simulation*, 14(1), p. 2.

Park, M., Lee, H. and Kim, H. (2021) 'Immersive simulation environments for safety awareness training in construction', *Journal of Computing in Civil Engineering*, 35(5), p. 04021052.

Park, J., Yoon, S., Choi, J. and Kang, D. (2021) 'Conversion pipeline from BIM to game engines for construction simulation', *Automation in Construction*, 127, p. 103697.

Pathak, D., Agrawal, P., Efros, A.A. and Darrell, T. (2017) 'Curiosity-driven exploration by self-supervised prediction', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 16–17.

Pedro, A., Le, Q.T. and Park, C.S. (2016) 'Framework for integrating safety into VR-based construction training', *Automation in Construction*, 62, pp. 53–66.

Ranaweera, M. (2024) 'Deep reinforcement learning with Godot game engine', *Electronics*, 13(5), p. 985. Available at: https://www.mdpi.com/2079-9292/13/5/985 (Accessed: 31 August 2025).

Ranaweera, M., Bhattarai, B., Lee, D. and Lee, G. (2024) 'Flexible scenario generation in Godot simulator', *arXiv preprint*, arXiv:2412.18408v1. Available at: https://arxiv.org/html/2412.18408v1 (Accessed: 31 August 2025).

Sacks, R., Eastman, C.M., Lee, G. and Teicholz, P. (2018) *BIM Handbook: A Guide to Building Information Modeling for Owners, Designers, Engineers, Contractors, and Facility Managers*. 3rd edn. Hoboken, NJ: Wiley.

Sacks, R., Perlman, A. and Barak, R. (2009) 'Construction safety planning using the safety activity theory model', *Journal of Construction Engineering and Management*, 135(8), pp. 700–708.

Sadeghi, M., Javadi, S., Taghaddos, H. and Hammad, A. (2019) 'Linking BIM and ABM for construction safety analysis', *Automation in Construction*, 107, p. 102933.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O. (2017) 'Proximal policy optimization algorithms', *arXiv preprint*, arXiv:1707.06347. Available at: https://arxiv.org/abs/1707.06347 (Accessed: 31 August 2025).

Shohet, I.M., Shapira, A., Weiss, P. and Gilboa, E. (2019) 'Feasibility of automated hazard detection in buildings using machine learning', *Automation in Construction*, 107, p. 102914.

Smith, D.K. and Tardif, M. (2009) *Building Information Modeling: A Strategic Implementation Guide for Architects, Engineers, Constructors, and Real Estate Asset Managers*. Hoboken, NJ: Wiley.

Sutton, R.S. and Barto, A.G. (2018) *Reinforcement Learning: An Introduction*. 2nd edn. Cambridge, MA: MIT Press.

Tavakolan, M. and Nasirzadeh, F. (2014) 'An agent-based model for evaluating construction projects' safety performance', *Automation in Construction*, 42, pp. 1–10.

Teizer, J., Cheng, T. and Fang, Y. (2013) 'Automation in construction safety: Review and future directions', *Automation in Construction*, 35, pp. 144–152.

Wang, X., Chong, H.Y. and Zhang, R. (2016) 'A framework for 4D BIM in construction safety management', *Automation in Construction*, 72, pp. 188–198.

Wang, X., Kim, M.J., Love, P.E.D. and Kang, S.C. (2014) 'Serious games for workplace safety: An application to electrical safety training', *Automation in Construction*, 39, pp. 129–144.

Wu, W., Yang, H. and Chew, D.A.S. (2014) 'A game engine-based simulation of construction safety for educational purposes', *Automation in Construction*, 39, pp. 32–41.

Xu, S., Lin, B. and Zou, P.X.W. (2023) 'Examining construction group's safety attitude resilience under major disruptions: An agent-based modelling approach', *Safety Science*, 161, p. 106071.

Xu, Y., Fang, D. and Huang, X. (2023) 'Examining construction group's safety attitude resilience under major disruptions', *Journal of Construction Engineering and Management*, 149(1), p. 04022148.

Yang, W., Tang, Y. and Wang, X. (2023) 'A reinforcement learning-enhanced ABM for dynamic construction safety simulation', *Journal of Computing in Civil Engineering*, 37(1), p. 04022056.

Zaman, R., Abdirad, H., Lee, G., Sacks, R. and Heaton, R. (2024) 'Towards an integrated framework for digital twins in construction safety training', *Automation in Construction*, 152, p. 104312.

Zhang, M., Chi, S. and Kim, H. (2015) 'BIM and agent-based modeling integration for safety-aware construction planning', *Advanced Engineering Informatics*, 29(3), pp. 406–421.

Zhang, H., Chi, S. and Lee, J.K. (2021) 'Integrating BIM and game engines for safety simulation in construction: A systematic review', *Journal of Construction Engineering and Management*, 147(7), p. 04021061.

Zhang, S., Teizer, J., Lee, J.K., Eastman, C.M. and Venugopal, M. (2013) 'Building Information Modeling (BIM) and safety: Automatic safety checking of construction models and schedules', *Automation in Construction*, 29, pp. 183–195.

Zhang, Y., Zhang, L. and Shen, Q. (2020) 'Real-time safety training using ABM and wearable sensors', *Automation in Construction*, 113, p. 103145.

Zhao, D., McCoy, A.P. and Bulbul, T. (2020) 'A framework for real-time interaction between BIM and simulation environments', *Journal of Computing in Civil Engineering*, 34(5), p. 04020044.

Zhou, W., Whyte, J. and Sacks, R. (2012) 'Construction safety and digital design: A review', *Automation in Construction*, 22, pp. 102–111.

Zhou, Y., Ding, L., Love, P.E.D. and Luo, H. (2013) 'A framework for integrating BIM and sensor data for real-time safety management', *Automation in Construction*, 29, pp. 183–195.

Zohdy, M., Omar, T. and McCabe, B. (2020) 'Agent-based evacuation simulation on scaffolding systems', *Automation in Construction*, 113, p. 103128.

Zolfagharian, M., Jelodar, H., Nair, R. and Love, P.E.D. (2023) 'SMARLA: Safety monitoring for deep reinforcement learning agents', *arXiv preprint*, arXiv:2308.02594. Available at: https://arxiv.org/abs/2308.02594 (Accessed: 31 August 2025).

Zolfagharian, S., Ma, X. and Love, P.E.D. (2023) 'Real-time safety monitoring using machine learning: A review and future directions', *Safety Science*, 161, p. 106057.

APPENDICES

APPENDIX 1: AICONTROLLER3D.GD - RECORDS SESSION START - LOG_TRAINING_START()

```
func log_training_start():
	var session_path := "user://training_session_log.csv"
	# Create log file with headers if not already there
	if not FileAccess.file_exists(session_path):
		var file := FileAccess.open(session_path, FileAccess.WRITE)
		file.store_line("Event,Episode,Timestamp,Details")
		file.close()
	var file := FileAccess.open(session_path, FileAccess.READ_WRITE)
	if file:
		file.seek_end()
		var timestamp = Time.get_datetime_string_from_system()
		file.store_line("TrainingStart,%d,%s,Batch RL training session started" %
[episode_count, timestamp])
		file.close()
	else:
		push_error("✕ Could not open session log for writing.")
```

APPENDIX 2: AICONTROLLER3D.GD - RECORDS END OF EACH EPISODE - LOG_EPISODE_END()

```
func log_episode_end(reason: String = "NormalEnd"):
	var session_path := "user://training_session_log.csv"
	var file := FileAccess.open(session_path, FileAccess.READ_WRITE)
	if file:
		file.seek_end()
		var timestamp := Time.get_datetime_string_from_system()
		file.store_line("EpisodeEnd,%d,%s,%s" % [episode_count, timestamp, reason])
		file.close()
	else:
		push_error("✕ Could not write episode end to session log.")
```

APPENDIX 3: AICONTROLLER3D.GD - TRACKS STEP-BY-STEP BEHAVIOUR: LOG_ACTIVITY()

```
func log_activity(step: int, pos: Vector3, vel: Vector3, phase: int, move: float, turn: float, step_reward: float, ledge_ratio: float):
	if not activity_log_initialized:
		var file := FileAccess.open(activity_log_path, FileAccess.WRITE)
		if file:
	file.store_line("Step,PosX,PosY,PosZ,VelX,VelY,VelZ,Phase,Move,Turn,Reward,Ledge Ratio")
			file.close()
			activity_log_initialized = true
		else:
			push_error("✗ Could not initialize episode summary CSV file.")
	var file := FileAccess.open(activity_log_path, FileAccess.READ_WRITE)
	file.seek_end()
	var line := "%d,%.2f,%.2f,%.2f,%.2f,%.2f,%.2f,%d,%.2f,%.2f,%.3f,%.2f" % [
		step,
		pos.x, pos.y, pos.z,
		vel.x, vel.y, vel.z,
		phase,
		move,
		turn,
		reward,
		ledge_ratio
	]
```