Proceedings of the 10th World Congress on Civil, Structural, and Environmental Engineering (CSEE 2025)
Barcelona, Spain - April, 2025
Paper No. ICSECT 182
DOI: 10.11159/icsect25.182

Automating BIM (IFC) Data Analysis Using LangGraph

Miral Selim[1], Khaled Nassar[2], Ossama Hosny[2]
[1]The American University in Cairo
AUC Avenue, New Cairo 11835, Egypt
selimiral@aucegypt.edu; knassar@aucegypt.edu
[2]The American University in Cairo
AUC Avenue, New Cairo 11835, Egypt
ohosny@aucegypt.edu

Abstract **-** This study introduces a novel methodology for automating the analysis of Building Information Modeling (BIM) data using LangGraph and integrating Google's Gemini Large Language Model (LLM) with IfcOpenShell. BIM, and specifically Industry Foundation Class (IFC) files, are widely used in the construction industry for representing and managing building data. However, analyzing this data effectively remains a significant challenge due to its volume and complexity. Additionally, analyzing BIM data typically requires knowledge of different BIM software depending on the application. This research addresses this challenge by creating a workflow that utilizes LangGraph's ability to develop different AI agents designed to handle tasks like extracting element data, analyzing spatial relationships, and categorizing risks based on predefined criteria, without the need for any BIM software at all. The integration of Gemini LLM provides advanced language-based reasoning and decision-making capabilities that allow the system to process complex queries, in human language, and provide valuable insights from the BIM data. As a proof-of-concept, four applications of the LangGraph methodology were created, providing significant insights regarding the strengths and limitations of this framework. The models were validated through hypothetical case studies and real-world applications, and responses were evaluated based on their accuracy, validity, and completeness, demonstrating the framework's effectiveness in analyzing BIM data in construction projects. However, the results also revealed limitations that can affect the system's performance in large-scale real-world applications. These findings suggest that while the proposed system shows great potential, further optimization is needed to enhance its usability and reliability in more complex and large-scale scenarios.

Keywords**:** Building Information Modelling, Automation, Large Language Models, Artificial Intelligence

1. Introduction

The construction industry has witnessed significant advancements in digital tools, with Building Information Modeling (BIM) emerging as a transformative technology. By providing a centralized repository for project information, BIM has enhanced collaboration, accuracy, and efficiency in design and construction workflows. Among the various standards for BIM, the Industry Foundation Class (IFC) format stands out as a widely adopted open standard for representing and exchanging building data. Despite its benefits, analyzing the complex and voluminous data within IFC files presents a persistent challenge, often requiring domain expertise and specialized software tools. Traditionally, BIM data analysis relies heavily on proprietary software, creating barriers such as steep learning curves, high costs, and limited interoperability. These challenges highlight the need for innovative solutions that can simplify data analysis while reducing dependency on traditional software. Recent developments in artificial intelligence (AI), particularly large language models (LLMs) like Google's Gemini, have shown promise in addressing complex reasoning and decision-making tasks. When combined with advanced frameworks like LangGraph and tools such as IfcOpenShell, AI presents an opportunity to revolutionize BIM data analysis.

2. Literature Review

Recent advancements in AI, particularly LLMs and Natural Language Processing (NLP), have been increasingly applied to BIM for tasks such as data visualization, access democratization, compliance checking, and automated data extraction. Alla, Osello, Rimella, and Stradiotto [1] developed a web-based IFC viewer aimed at improving visualization and user interaction within BIM workflows. Their system integrated synchronized model alignments and cross-sectional

visualizations while leveraging NLP-powered conversational agents to enable intuitive user commands. The study demonstrated how AI can enhance accessibility through cost-effective, web-based solutions, making BIM more user-friendly for non-technical stakeholders. Similarly, Massafra, Coragila, Predari, and Gulli [2] explored the integration of LLMs in BIM to improve model accessibility. Their research employed a knowledge graph structure where BIM data was stored as nodes and edges, allowing ChatGPT to process textual queries and provide analysis. The user-friendly graphical interface further simplified interactions, making BIM data analysis more intuitive for non-expert users.

A growing body of research has also investigated the role of LLMs in Automated Compliance Checking (ACC) within the architecture, engineering, and construction (AEC) industry. Iversen, Huang, and Merschbrock [3] examined LLMs for BIM-based validation of natural language building regulations. Their artifact incorporated four modules: rule analysis, BIM model extraction, compliance validation, and report generation. The system achieved F1 scores of 97% for regulation classification and 100% for rule dependency identification, showcasing LLMs' potential to reduce manual labor and enhance regulatory transparency. Ying and Sacks [4] advanced this approach by developing an autonomous LLM agent for ACC. Using Retrieval-Augmented Generation (RAG) and the ReAct framework, their system retrieved design requirements, planned and executed checks, and generated compliance reports while also supporting model revisions based on rule-checking results. Their findings highlighted LLMs' ability to streamline compliance validation and automate complex design tasks. Chen, Lin, Jiang, and An [5] further expanded on AI-driven ACC by integrating deep learning models and ontology-based knowledge frameworks, thereby improving accuracy and efficiency in regulatory compliance verification.

Beyond compliance, AI has also been applied to BIM data structuring and querying. Chu, Wu, and Lei [6] addressed challenges in querying construction data by developing IFC-Graph, a model-driven approach that converts IFC data into labeled property graphs. By mapping IFC concepts to graph structures, their methodology enabled efficient data retrieval and improved queryability compared to traditional relational databases. Wang, Issa, and Anumba [7] took a different approach by designing an NLP-based query-answering system for BIM. Their system used three modules—Natural Language Understanding, Information Extraction, and Natural Language Generation—to interpret user queries, extract IFC data, and generate natural language responses. A prototype tested on seven BIM models and 127 queries achieved 81.9% accuracy, demonstrating NLP's potential in making BIM data more accessible. Similarly, Dawood, Siddle, and Dawood [8] combined NLP and IFC data models to identify and manage design changes. Their web-based platform visualized model differences and employed NLP to detect modifications across design iterations, assisting designers and 3D modelers in tracking project updates.

A Systematic Literature Review by Du, Hou, Zhang, Tan, and Mao [9] assessed the readiness of BIM data for AI applications. By analyzing 93 studies from SCOPUS and WoS, the authors identified eight common data types, two data management frameworks, and four primary data conversion methods used in BIM. Their findings suggest that while AI integration in BIM is advancing, data readiness remains at an intermediate stage, requiring further optimization for seamless AI adoption.

One promising AI framework for handling complex, multi-step workflows in BIM is LangGraph. LangGraph is designed to structure AI workflows using a graph-based approach, where tasks are represented as nodes and execution follows a dynamic sequence [10]. Unlike traditional linear systems, LangGraph supports cyclic computation, allowing AI agents to revisit and refine previous steps based on evolving data and real-time decisions. This makes it particularly useful for applications requiring iterative reasoning, adaptive decision-making, and continuous interaction with external data sources. LangGraph operates using a stateful graph, where dynamic memory is updated throughout execution. Each node can access this state, ensuring informed decision-making at every stage. Nodes execute specific tasks, such as processing inputs, interacting with external systems, or making decisions, while edges define the execution flow and enable conditional branching based on real-time data. This flexibility allows AI agents to navigate complex, multi-step workflows that adapt dynamically to changing conditions.

LangGraph has been successfully implemented in various fields to develop advanced AI-based applications. Easin, Sourav, and Tamás [11] utilized LangGraph to create an intelligent personalized assistant for digital banking, integrating a multi-agent framework with Chain of Thoughts prompting. By mapping user interactions to structured nodes, the

system efficiently handled banking services such as fund transfers, bill payments, and savings management. Similarly, Liu, Kang, and Han [12] leveraged LangGraph to optimize Retrieval-Augmented Generation for processing complex automotive industry documents, developing a self-RAG agent that improved retrieval accuracy and context compression in offline PDF chatbots. Their approach outperformed naive RAG baselines, highlighting LangGraph's effectiveness in industry-specific intelligent document processing.

De Alba, Kobayashi, and Cabello [13] applied LangGraph to Space Domain Awareness, integrating it with a space scene simulator to enhance accessibility for configuring complex simulations. LangGraph structured the system's modular design, enabling natural language interactions for precise simulation configurations. Experiments demonstrated its robustness, achieving 70.41% accuracy in mapping user inputs to valid configurations. In healthcare innovation, Aikins and Khansa [14] employed LangGraph to optimize AI-driven patent analysis and innovation management. The framework structured and secured multi-agent workflows, ensuring scalable operations while addressing privacy concerns. Lastly, Timms, Langbridge, and O'Donncha [15] utilized LangGraph for anomaly detection in maritime shipping, orchestrating an agentic system that analyzed environmental conditions and system dynamics for predictive maintenance. By managing multi-agent workflows, LangGraph enabled efficient routing of tasks such as anomaly detection, asset health forecasting, and maintenance scheduling, marking a significant advancement in operational decision-making.

3. Methodology

The methodology involves LLM setup and initialization, IFC data parsing and preparation, querying Gemini LLM, and processing of results. For the LLM setup and initialization, an API key for Google Gemini was obtained through the Google API Console, and authentication was configured in the Python environment. IfcOpenShell was installed and configured to parse and manipulate IFC files, which store building data. LangGraph was then set up to orchestrate the workflow as a multi-agent system. For IFC data parsing and preparation, IfcOpenShell was utilized to load and parse IFC files. Building data, including geometry, spatial relationships, and properties, was extracted and preprocessed to ensure compatibility with the LLM. The extracted data was converted into a structured text format or prompts that could be effectively processed by the LLM. To query Gemini LLM, the preprocessed data was sent to the LLM through LangGraph. The LLM was tasked with handling challenging queries, such as analyzing spatial relationships, identifying compliance issues, or proposing design modifications. Specific prompts tailored to each task were employed to generate relevant insights. As for processing results, the outputs from the LLM were processed to produce actionable recommendations or insights, such as identifying conflicts in design, compliance issues, or optimization suggestions.

4. Applications of Proposed Model

Four proof-of-concept applications were developed based on the above methodology: Element Analysis Agent, Solid Slab Area Analysis Agent, Footing Area Analysis Agent, and Multi-Agent Depth & Proximity Analysis Workflow.

The Element Analysis Agent is designed to extract all data relevant to a specific element in the BIM model. By allowing users to query specific elements and retrieve all related information, it offers precise insights into individual building components, streamlining workflows and enhancing decision-making. Fig. 1 shows the Element Analysis Agent's LangGraph.

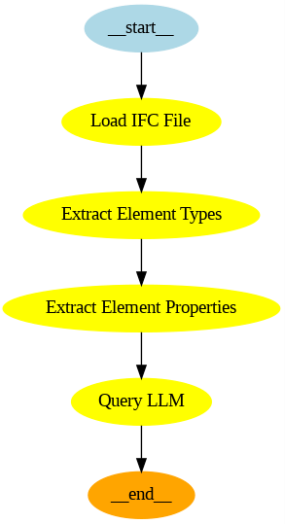


Fig. 1: Element Analysis Agent LangGraph

The graph starts at the initialization node, where the process is prepared for execution. From there, it transitions to the first operational node, which is responsible for loading the IFC file. At this stage, the application opens the specified file using IfcOpenShell and loads its data into memory. If the file is successfully loaded, the information is stored in the graph's state for further analysis; otherwise, appropriate error messages are generated, and the workflow halts. Once the file is loaded, the workflow proceeds to the next node, where the application identifies all available element types in the IFC model. This involves scanning the file to categorize its contents, such as columns, walls, beams, or other building elements. The extracted types are stored in the graph state, providing the basis for the next analytical step. Following this, the workflow moves to the node dedicated to extracting properties of a specific element type selected by the user. This step retrieves detailed information such as the Global IDs, names, and specific attributes of the selected elements. The extracted data not only offers an in-depth understanding of the elements but also serves as the context for the next stage. After the properties are extracted, the workflow transitions to the node responsible for querying the LLM. At this stage, the LangGraph uses the previously extracted element data along with a user-provided query to interact with the LLM. For example, the user might ask for detailed insights about column elements or inquire about the number of beam elements in the model. The LLM processes the input data and query in context, generating an AI-driven response. This response is stored in the graph state and provides valuable insights derived from the combination of BIM data and the LLM's interpretive capabilities. The LangGraph concludes at the final node, where all results are collated, including the extracted data and the AI-generated insights. The end node marks the completion of the process, and the output is either displayed to the user or logged for further use. This structured and sequential process ensures that the IFC data is analyzed comprehensively, enabling professionals in architecture, engineering, and construction to derive meaningful insights.

The Solid Slab Area Analysis Agent analyzes slab elements in an IFC file to identify slabs that exceed a specified area threshold. Fig. 2 shows the Solid Slab Area Analysis Agent’s LangGraph.

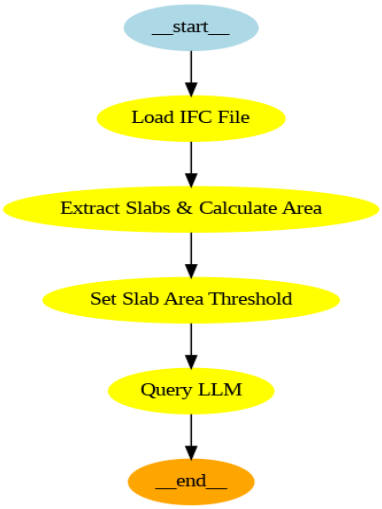


Fig. 2: Solid Slab Area Analysis Agent LangGraph

The workflow begins with the initialization of the process, where the IFC file is opened and parsed using IfcOpenShell, and its contents are loaded into memory. This step establishes the foundational data required for slab analysis. If the file cannot be loaded, appropriate errors are logged, and the process halts. Next, the application extracts all IfcSlab elements from the model, filtering out foundation slabs and pile caps based on their names. The extracted slabs areanalyzed to compute their areas by processing their geometry. Specifically, slabs with rectangular profiles (IfcRectangleProfileDef) are evaluated by multiplying their dimensions to determine their area in square meters. Following the area calculations, the LangGraph prepares a summary of all extracted slabs along with their calculated areas and the predefined threshold (set to 36 m^2 in this application). This information is then sent to the AI model for evaluation. The Gemini LLM processes the provided data and generates insights, warnings, or recommendations related to slabs that exceed the threshold. These insights may focus on dimensions, placement, or structural considerations, offering valuable guidance based on the slab properties and the defined threshold. Finally, the process concludes with the output of slabs exceeding the area threshold and the AI-generated insights. These results assist in identifying potential design issues or structural concerns, providing actionable recommendations for further evaluation or modification.

The Footing Area Analysis Agent differs conceptually from the Solid Slab Area Analysis Agent by focusing on the relationship between columns and their supporting footings, rather than analyzing slabs independently for area thresholds. Accordingly, it analyzes the geometric properties of one element in comparison to the geometric properties of another element with which a relationship exists, and not geometric properties vs a user-defined threshold. The key distinction lies in this interdependency: the footing area must be proportionally sufficient to support the column, with specific warnings generated if this proportionality is not met (refer to Fig. 3).

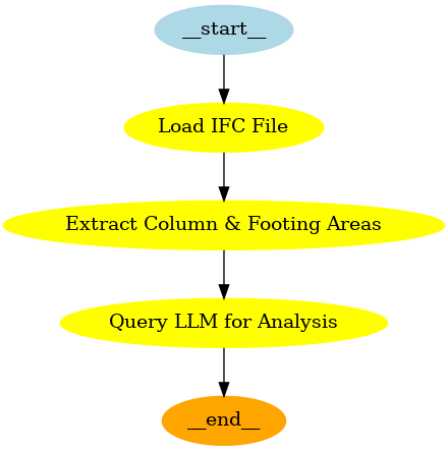


Fig. 3: Footing Area Analysis Agent LangGraph

The workflow begins by loading the specified IFC file, extracting the dataset into memory. This foundational step ensures that all required data for column and footing analysis is accessible. If the file cannot be loaded, the process halts with logged errors. Once the data is loaded, LangGraph proceeds to extract column and footing elements from the model using IfcOpenShell. For columns and footings with rectangular profiles, LangGraph calculates their areas by multiplying the dimensions and converting the results to square meters. After the area extraction, the compiled results, including column and footing names and their calculated areas, are sent to the Gemini model. The AI model processes this information and evaluates whether the footing areas are proportionally sufficient to support the corresponding columns (at least 2.5 times the column area). The AI generates insights, warnings, or recommendations based on the provided data, offering a detailed assessment of potential structural concerns or design issues. The process concludes with the output of the AI-generated recommendations and flagged footing-column issues, if any. This ensures that potential design flaws, such as insufficient footing dimensions, are identified and addressed early in the design process.

The Multi-Agent Depth and Proximity Analysis Workflow uses a LangGraph with a Supervisor Architecture setup designed to analyze the structural details of an IFC file, with a specific focus on spatial relationships and differential settlement risks of foundation elements. At the core of the system is the top-level Supervisor Node, which orchestrates the entire workflow by determining which agents to use and in what order based on the analysis requirements. This supervisory control ensures that the process is both dynamic and adaptive, allowing for tailored execution of tasks. In this application, the model analyzes structural foundations based on height and proximity thresholds simultaneously, categorizes them into 3 risk groups, and responds to prompts in this context. Figure 4 shows the LangGraph of the Multi-Agent Depth & Proximity Analysis Workflow.

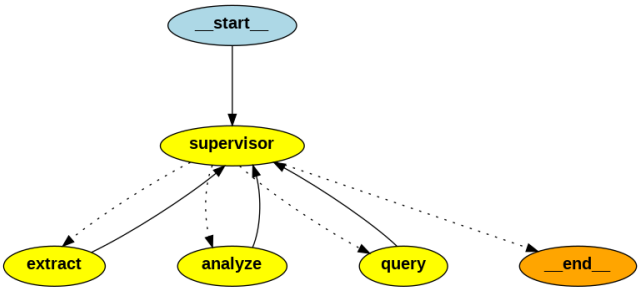


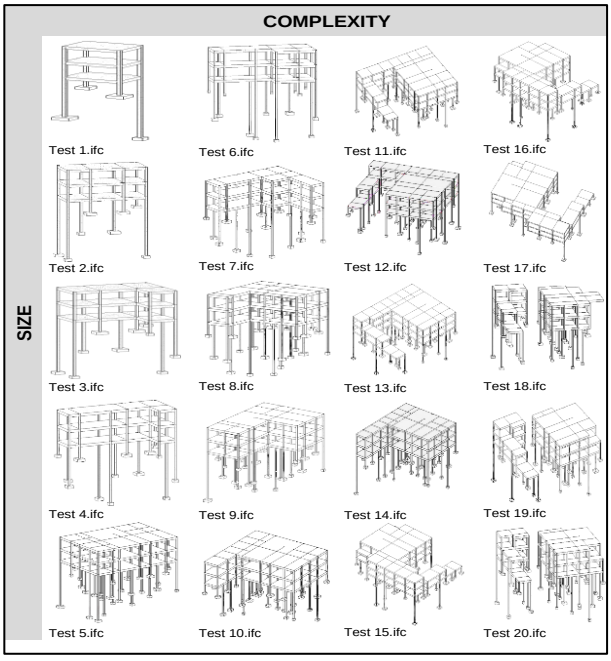
Fig. 4: Multi-Agent Depth & Proximity Analysis Workflow LangGraph

The LangGraph begins with a Unit Check and Conversion Node, which ensures that all measurements within the IFC file are in metric S.I. units. This initial step establishes consistency for subsequent computations. The supervisor

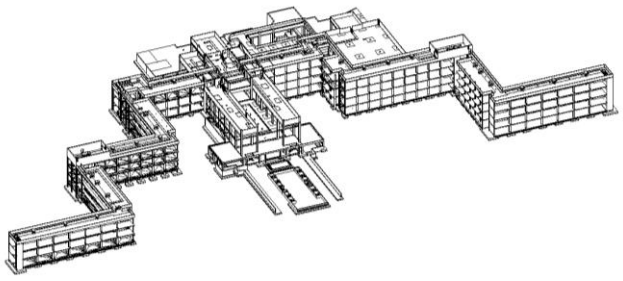
then activates the Element Data Extraction Agent, which uses IfcOpenShell to parse the IFC file to extract crucial details about structural elements, including their geometries, spatial positions, and foundation depths. The extracted information is is formatted into a structured dataset for further processing. Next, the supervisor triggers the Proximity Analysis Agent, which calculates the clear distances between foundation elements. This agent evaluates whether the elements maintain the the minimum required spacing to mitigate potential risks. The supervisor's logic determines when to deploy this agent based based on the spatial data provided by the extraction process. Once proximity data is obtained, the supervisor decides to activate the Analysis Agent, which applies predefined thresholds to assess differential settlement risks. These thresholds include parameters for depth differences (h) and distances (d), with user-defined ranges categorizing risk levels into Low, Medium, and High. For example, Low Risk corresponds to $h < 5m$ and $d > 3m$, Medium Risk to $5 < h < 7m$ and $2 < d < 3m$, and High Risk to $h > 7m$ and $d < 2m$. The supervisor allows for adjustments to these thresholds, enabling a customizable and precise risk assessment. Once all analyses are completed, the supervisor consolidates the results, synthesizing outputs from the individual agents into a comprehensive dataset. The final step involves activating the LLM Query Node, which interprets the combined data and provides insights or recommendations using a language model. This includes generating observations about foundation risks and suggesting potential mitigation strategies.

5. Validation

The validation strategy consists of two parts: first, each of the developed models was tested using 20 hypothetical case studies and the results were tabulated and analyzed to ensure that the models meet their intended use. The test case studies were sorted in order of size and complexity to test if the models' performance is affected by the size and complexity of the BIM model being processed; refer to Fig. 5 (a). Next, each model was tested using an actual case study of a real-life project to test the functionality of the system and evaluate its performance in a real-life scenario. The construction project on which the models were tested was a large, irregularly shaped hotel building located in a coastal region.; refer to Fig. 5 (b).



(a) Hypothetical Case Studies



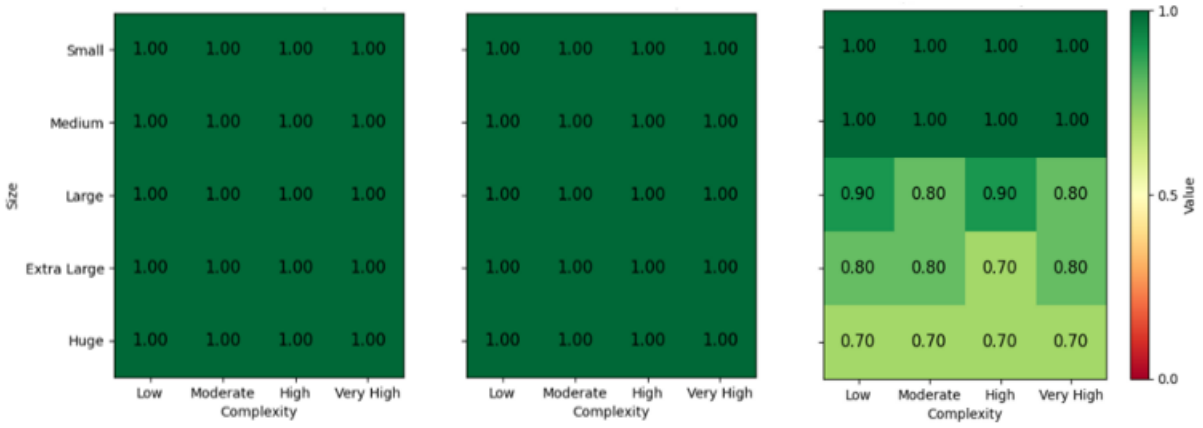
(b) Real-Life Case Study

Fig. 5: Test Models for Validation

The performance of each LangGraph model was assessed based on accuracy, validity, and completeness of the response provided. Accuracy measures how closely the model's responses align with the true values. Validity evaluates whether the model's results are relevant and appropriately addresses the specific task or question at hand. Completeness assesses whether the model provides a full and comprehensive response, covering all necessary aspects of the problem or query.

6. Results & Discussion

The results of testing the models were collected and compared to manually obtained data, and a score of 0 to 1 (0% to 100%) was assigned. For the Element Analysis Agent, the test files were uploaded, and the model was asked "Can you provide details on columns elements?" The model's responses were evaluated as shown in Fig. 6.

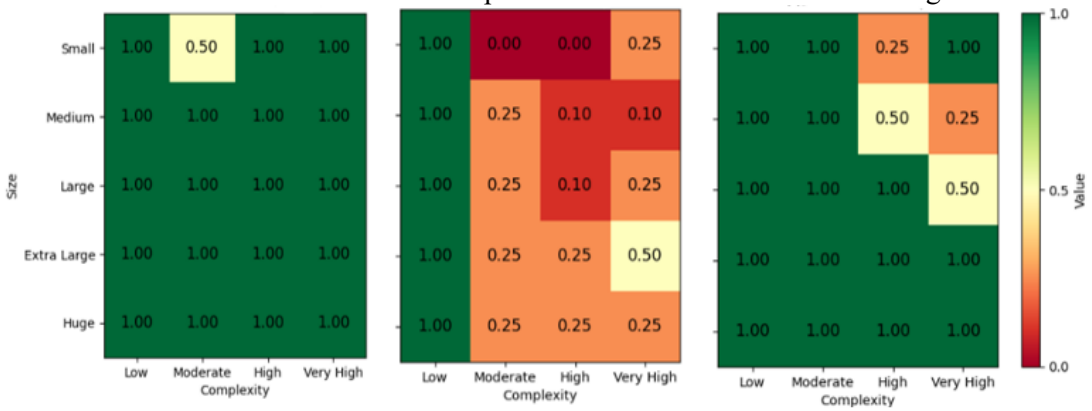


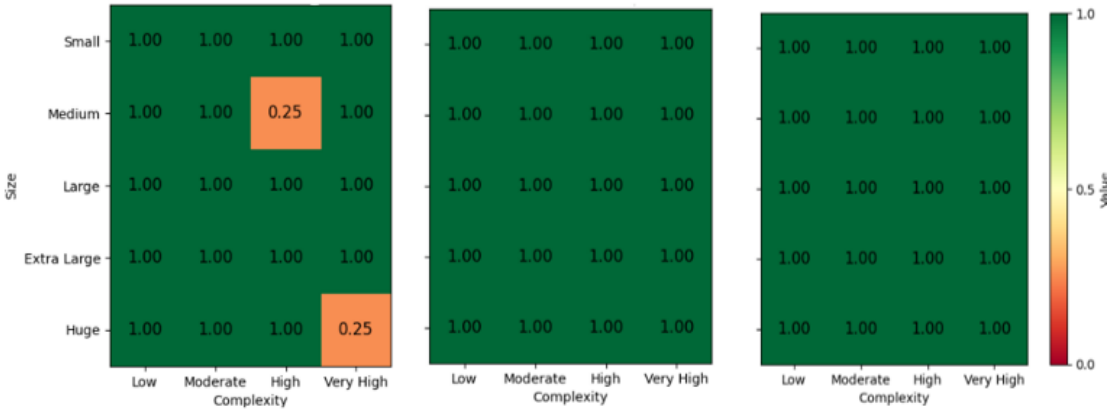
Accuracy Validity Completeness

Fig. 6: Element Analysis Agent Results

The model demonstrated excellent performance in analyzing IFC files, achieving 100% accuracy across all file sizes and complexity levels by correctly identifying and counting IfcColumn elements. Its response validity was also perfect (100%), providing relevant and context-sensitive information, ranging from basic details to advanced inferences when necessary. While response completeness averaged 86.5%, with detailed outputs for smaller models and summarized data for larger ones, the responses remained informative and valuable. A real-life case study further validated the model's effectiveness, as it successfully identified 707 column elements, verified unique Global IDs and Names, and accurately summarized key properties. Future improvements could enhance completeness, particularly for larger BIM models.

For the Solid Slab Area Analysis Agent, the test files were uploaded, and the model was asked "Analyze the slabs and provide any issues with their areas." The model's responses were evaluated as shown in Fig. 7.





Accuracy Validity Completeness

Fig. 7: Solid Slab Area Analysis Agent Results

The model achieved high accuracy (98%) in identifying slabs exceeding the 36 m² threshold, but it encountered an isolated threshold misapplication error in 'Test 6.ifc.' While it correctly flagged slabs based on area, its reasoning was highly inconsistent, with a response validity score of only 40%. The model often applied incorrect thresholds, provided flawed or nonsensical justifications, or omitted reasoning altogether, highlighting the need for human oversight. Response completeness averaged 88%, with sufficient details in simpler BIM models but inconsistencies in complex scenarios. In a real-life case study, the model correctly identified slabs exceeding the threshold but failed to provide meaningful insights, indicating its limitations in handling complex projects. Improvements should focus on ensuring correct threshold application, refining error handling, and integrating a Human-in-the-Loop validation feature to enhance reasoning reliability.

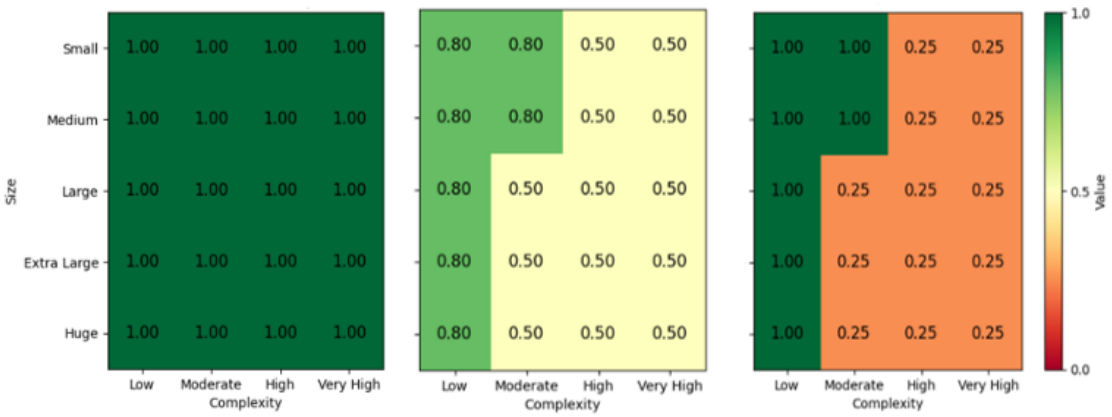
For the Footing Area Analysis Agent, the test files were uploaded, and the model was asked "Analyze the column and footing areas and raise any issues." The model's responses were evaluated as shown in Fig. 8.

Accuracy Validity Completeness

Fig. 8: Footing Area Analysis Agent Results

The model accurately identified all foundation and column instances, achieving a 93% accuracy in flagging undersized foundations. However, it incorrectly flagged some foundations as "too large" in two cases, suggesting minor logical refinements are needed. Response validity scored 100%, as the model provided relevant insights and logical recommendations, demonstrating advanced reasoning beyond simple calculations. Response completeness was also 100%, as it consistently delivered comprehensive analyses, listing foundation-to-column area ratios and offering design improvements. In a real-life case study, the model successfully analyzed foundation sizing but encountered network-related limitations (HTTP 429 errors), preventing further assessments. This highlights the need for optimized querying and better error handling for large-scale applications.

As for the Mult-Agent Depth & proximity Analysis Workflow, the test files were uploaded, and the model was asked "Please confirm the number of high-risk foundations" followed by "Provide recommendations based on this data." The model's responses were evaluated as shown in Fig. 9.



Accuracy Validity Completeness

Fig. 9: Footing Area Analysis Agent Results

The model accurately identified all high-risk foundations in every test case, achieving 100% accuracy in risk detection. It successfully classified foundation risks and provided mostly relevant recommendations, scoring 60% in response validity. However, in cases 'Test 8.ifc' to 'Test 20.ifc', network-related errors prevented the model from generating recommendations, impacting response completeness, which averaged 51%. The real-life case study confirmed that connectivity issues were likely triggered by the high number of design issues rather than model size or complexity. This suggests that the model can handle large BIM files but struggles when excessive risk cases increase response times.

The validation process across multiple cases has revealed important insights into the performance of the applications of the proposed approach for analyzing IFC data using LLM through LangGraph. The developed models (focused on element analysis, slab area analysis, footing area analysis, and multi-agent depth and proximity analysis) demonstrated strong potential for real-world applications, while also highlighting specific limitations that could impact their use in practical scenarios. Fig. 10 summarizes the models' performance across the 3 evaluation criteria. The heatmap shows that most of the models achieved impressive accuracy and completeness, showcasing their effectiveness in analyzing BIM data and providing actionable insights. These strengths demonstrate the potential of LangGraph to enhance BIM-based decision-making and project outcomes. However, some models faced challenges in validity, indicating opportunities for improvement in reasoning and contextual understanding. This variation in scores emphasizes the need for ongoing refinement to further enhance the models' capabilities and improve their reliability.

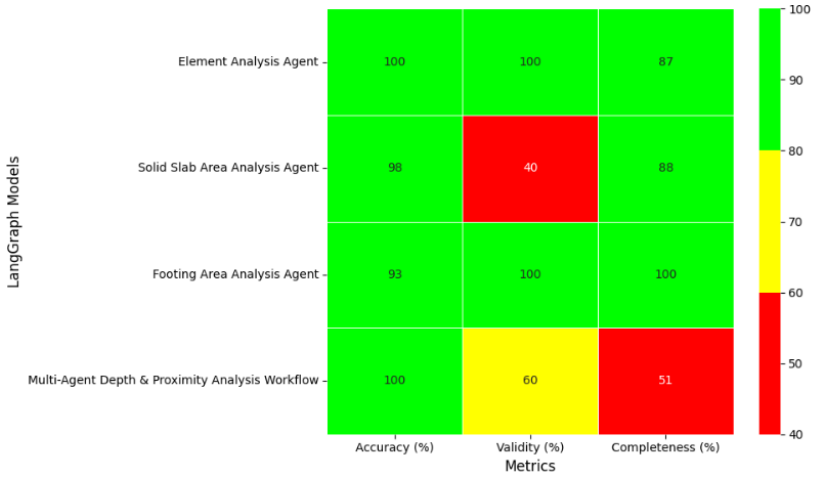


Fig. 10: Performance Heatmap of the LangGraph Models

The LangGraph method offers high accuracy in identifying structural concerns and generating actionable insights, aiding aiding decision-making and improving project outcomes. However, limitations in reasoning validity and response completeness, especially in complex scenarios, affect its reliability. While well-suited for small to medium-scale projects, projects, human verification remains essential for accuracy. Future improvements should focus on refining reasoning and and completeness to enhance applicability in larger projects. Despite its challenges, LangGraph shows strong potential for BIM analysis, and further development could significantly impact project success.

7. Conclusion

This study presents a novel methodology for automating the analysis of Building Information Modeling (BIM) data using LangGraph and Google's Gemini Large Language Model (LLM) in combination with IfcOpenShell. By leveraging a multi-agent system, the proposed framework enables efficient querying and analysis of IFC data without reliance on traditional BIM software. The research demonstrates the feasibility of using AI-driven approaches to extract, analyze, and interpret BIM data, providing valuable insights for construction professionals. The validation of the developed models across both hypothetical case studies and a real-life project confirmed their effectiveness in analyzing BIM data, particularly in element extraction, slab and footing analysis, and spatial risk assessment. The models exhibited high accuracy and completeness, successfully identifying critical design issues and generating actionable recommendations. However, certain limitations were observed, particularly in reasoning validity and response completeness when handling complex queries or large-scale models. Network-related constraints also impacted on the system's ability to process extensive datasets in real-world applications. While the findings highlight the potential of LLM-based BIM analysis, they also underscore the need for further optimizations. Enhancing reasoning accuracy, improving response completeness, and implementing robust error-handling mechanisms will be crucial in refining the system's reliability. Additionally, integrating Human-in-the-Loop validation and optimizing the framework for large-scale projects can enhance its practical applicability. Despite these challenges, the proposed methodology represents a significant step forward in automating BIM data analysis. With further advancements, AI-driven BIM analysis tools have the potential to improve decision-making, streamline workflows, and enhance project outcomes in the construction industry.

References

[1] S. Alla, F. Osello, P. Rimella, and L. Stradiotto, "A web-based IFC viewer integrating NLP-powered conversational agents for BIM workflows," *Automation in Construction*, vol. 154, p. 105012, 2024.

[2] L. Massafra, A. Coragila, G. Predari, and R. Gulli, "Enhancing BIM model accessibility through large language models and knowledge graphs," *Advanced Engineering Informatics*, vol. 58, p. 102168, 2024.

[3] E. Iversen, W. Huang, and C. Merschbrock, "Leveraging LLMs for automated compliance checking of building regulations in BIM," *Journal of Information Technology in Construction*, vol. 29, pp. 127–142, 2024.

[4] J. Ying and R. Sacks, "An autonomous LLM agent for automated compliance checking using retrieval-augmented generation and the ReAct framework," *Journal of Building Engineering*, vol. 78, p. 107632, 2024.

[5] Y. Chen, X. Lin, Y. Jiang, and H. An, "Deep learning-based AI framework for ontology-driven automated compliance checking in BIM," Automation in Construction, vol. 156, p. 105456, 2024.

[6] S. Chu, J. Wu, and L. Lei, "IFC-Graph: A model-driven approach for efficient querying of construction data using property graphs," Computers in Industry, vol. 152, p. 104592, 2024.

[7] Y. Wang, R. Issa, and C. Anumba, "Natural language query-answering system for BIM using NLP and information extraction," Journal of Computing in Civil Engineering, vol. 38, no. 2, p. 04023025, 2024.

[8] N. Dawood, J. Siddle, and H. Dawood, "An NLP-based approach for detecting and managing design changes in IFC models," Advanced Engineering Informatics, vol. 59, p. 102184, 2024.

[9] J. Du, L. Hou, C. Zhang, Y. Tan, and X. Mao, "BIM data readiness for AI applications: A systematic literature review," Automation in Construction, vol. 155, p. 105321, 2024.

[10] B. Lang and M. Graph, "LangGraph: A graph-based approach for structuring AI workflows in BIM applications," Artificial Intelligence in Engineering, vol. 65, p. 101453, 2024.
[11] M. Easin, A. Sourav, and G. Tamás, "A LangGraph-based multi-agent system for intelligent banking assistants," Expert Systems with Applications, vol. 231, p. 120994, 2024.
[12] Y. Liu, T. Kang, and D. Han, "Self-RAG: A LangGraph approach for optimizing retrieval-augmented generation in automotive document processing," Knowledge-Based Systems, vol. 273, p. 110295, 2024.
[13] A. De Alba, T. Kobayashi, and C. Cabello, "LangGraph in Space Domain Awareness: Enhancing simulation accessibility through AI-driven modular workflows," Aerospace Science and Technology, vol. 143, p. 108152, 2024.
[14] K. Aikins and L. Khansa, "AI-driven patent analysis using LangGraph for scalable multi-agent workflows in healthcare innovation," Artificial Intelligence in Medicine, vol. 134, p. 104371, 2024.
[15] P. Timms, D. Langbridge, and P. O'Donncha, "Anomaly detection in maritime shipping using LangGraph-based multi-agent workflows," Ocean Engineering, vol. 298, p. 114209, 2024.