

Article

Development of an RAG-Integrated Agentic BIM System for Intelligent Railway Maintenance

Minjae Jeon ¹, Yonggun Kim ¹ and Seok Kim ^{2,*}

¹ Department of Railroad Convergence System, Korea National University of Transportation, Uiwang-si 16106, Republic of Korea; minjae010829@a.ut.ac.kr (M.J.); xpdlv090100@ut.ac.kr (Y.K.)

² Department of Railroad Infrastructure Engineering, Korea National University of Transportation, Uiwang-si 16106, Republic of Korea

* Correspondence: kimseok@ut.ac.kr

Highlights

What are the main findings?

- Developed an Agentic BIM framework integrating Model Context Protocol (MCP) and Retrieval Augmented Generation (RAG) for autonomous railway maintenance decision-making.
- Demonstrated a cognitive visualization system that autonomously judges maintenance priorities based on guidelines and applies a color-based visual warning loop directly to BIM objects.

What are the implications of the main findings?

- Eliminates human errors and quality variances stemming from engineer proficiency by replacing manual data conversion and complex programming with natural language-based orchestration.
- Establishes a new paradigm for intelligent infrastructure management by evolving static 3D models into active autonomous operating systems where judgment and execution are fully unified.

Abstract

Modern railway maintenance is transitioning toward a condition-based maintenance system to stably operate the core infrastructure of sustainable smart cities. However, technical limitations remain in manually converting and analyzing massive amounts of inspection data into Building Information Modeling (BIM) objects. This causes information delays and technical severance in data-driven smart-city infrastructure. To address these challenges, this study proposes an Agentic BIM framework that integrates Large Language Model (LLM), Model Context Protocol (MCP), and Retrieval Augmented Generation (RAG) technologies. The proposed methodology standardizes the control channel between the LLM and BIM software through a central MCP server, while securing the accuracy of engineering judgments by utilizing the RAG pipeline to reference national railway-track-maintenance guidelines. System validation results demonstrated that geometric inspection data, including gauge and alignment, were automatically generated as BIM objects without human intervention. Furthermore, the maintenance grades and deadlines for sections exceeding thresholds were immediately highlighted within the model as visual attributes. Consequently, this framework proves an autonomous decision-making system that organically links inspection data with maintenance regulations. By transforming static, manual-labor-centered maintenance workflows into intelligent

Academic Editors: Isam Shahrouh and Pierluigi Siano

Received: 2 June 2026

Revised: 31 July 2026

Accepted: 3 August 2026

Published: 4 August 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\) license](https://creativecommons.org/licenses/by/4.0/).

automated models, it increases the efficiency of railway infrastructure management while providing a scalable technical foundation for overall asset management of future smart-city infrastructure.

Keywords: Large Language Model; Model Context Protocol; Retrieval Augmented Generation; Building Information Modeling

1. Introduction

Modern railway infrastructure, as a core mobility network constituting sustainable smart cities, requires higher maintenance precision and asset operation efficiency as passenger and logistics transportation accelerate toward high-speed and high-density operations [1]. As a critical national infrastructure facility, the rail network is highly sensitive to minor geometric deformations in rail alignment, such as gauge, cant, and track irregularities, which directly impact train running safety and ride comfort [2]. If rail irregularities intensify, they act as major impediments to train operations. Therefore, continuously quantifying and managing the geometric soundness of the track is essential [3]. Accordingly, the traditional manual inspection system [4,5] is shifting toward a condition-based maintenance system due to the advancement of high-precision sensors and data analysis technologies, allowing the accumulation of track conditions into continuous numerical values and the prediction of degradation trends [6,7].

The demand for the intelligence of digital infrastructure and spatial information constituting the smart-city ecosystem is increasing. Accordingly, computational research has recently been conducted in the architecture, engineering, and construction sector to automatically convert and optimize complex physical and engineering data into semantic digital models through algorithms. Maksound et al. [8] proposed a framework to automatically generate and optimize multiple architectural mass alternatives through a generative design technique utilizing the EvoMass algorithm. In parallel, continuous attempts have been made to introduce Building Information Modeling (BIM) [9,10] as the central axis of intelligent digital workflows. BIM serves as a platform to establish a maintenance ecosystem by integrating and managing shape and attribute information throughout the entire lifecycle of facilities [11].

However, these technological advancements do not directly lead to the complete automation of intelligent railway maintenance decision-making. In current practices, a technical disconnection exists in projecting and analyzing massive inspection data collected in a one-dimensional numerical array into a three-dimensional spatial model [12,13]. This is because engineers must manually interpret the irregularity values embedded in discretized points generated at regular intervals and map them individually onto object attributes within the BIM environment [14]. Consequently, despite securing quantitative inspection data, an automation ecosystem that transforms this data into intuitive 3D objects and links it with maintenance regulations is still absent [15]. This fragmented data integration process prevents dynamic field data from being organically reflected within BIM, causing information delays and data disconnection [16]. This acts as a major limiting factor in constructing a smart city infrastructure management system that needs to operate organically based on data. As a result, current BIM remains focused on the design and construction phases, remaining confined to the limitations of a static data repository without organically combining with field-inspection data during the maintenance phase [17].

Recently, the advancement of LLM (Large Language Models) has presented a new solution called Agentic AI, moving beyond simple information retrieval to independently perceive problems and utilize external tools [18]. However, two major limitations exist in

integrating LLM directly into railway engineering workflows. First, a lack of interoperability caused by proprietary closed software architectures impedes seamless communication. Second, the phenomenon of hallucination makes it difficult to guarantee the accuracy of engineering data [19].

To address these limitations, this study introduces a framework combining MCP (Model Context Protocol) and RAG (Retrieval Augmented Generation) technologies. First, the interoperability issue is resolved through MCP, a latest protocol that standardizes data exchange between AI agents and professional engineering tools [20]. Through this, the agent secures a pathway to access internal APIs of complex Building Information Modeling software such as Autodesk Revit. This enables the agent to go beyond a simple chatbot and function as an autonomous actor that reads and writes data directly within the Building Information Modeling environment [21]. Second, the hallucination phenomenon causing errors in engineering judgment is resolved by constructing a RAG pipeline [22]. Instead of generating arbitrary parameters or making undocumented judgments, the agent is designed to mandatory search and reference a knowledge base containing national standards, specifically the Railroad Track Maintenance Guidelines [23]. By controlling the state diagnosis and threshold-based visualization of the agent to be grounded in validated engineering criteria, the accuracy and reliability of the decision-making results are firmly secured.

The core purpose of this study is to demonstrate an automated workflow where field-inspection data directly leads to rapid and consistent maintenance decision-making without manual processing by establishing an Agentic BIM autonomous diagnosis framework centered on an MCP server as a central integration hub. The overall process of this study consists of four steps as shown in Figure 1. First, when the collected ① rail geometric inspection data is inputted, ② the LLM operates based on it. At this time, the LLM agent structures, perceives, and processes the collected data. Subsequently, the data is delivered straight to the Autodesk Revit environment, a Building Information Modeling software, through the central MCP server. Through this, ③ automated modeling is executed, where inspection data values from the field are inputted and updated as parameters of Building Information Modeling objects without manual intervention by engineers. After the data is reflected in the model, the process moves to ④ automated visualization, which displays the updated attribute data on the viewer. In this step, RAG technology is utilized to reference the Railroad Track Maintenance Guidelines [23] to determine and visualize anomalies and risk levels of the data inputted into the model. Consequently, by combining LLM and Building Information Modeling through the medium of MCP and RAG, this study overcomes the manual data processing limitations of conventional maintenance systems. Furthermore, the Agentic BIM framework presented in this study can be expanded as a foundational technology for intelligent models that support the digitization and decision-making of various core infrastructure assets within smart cities.

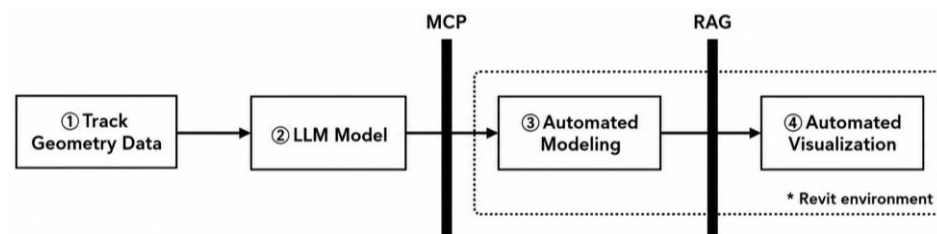


Figure 1. Agentic BIM system for intelligent railway maintenance process.

2. Related Work

2.1. Rail Parameter Extraction and BIM Data Integration Methodologies

To maximize the operational efficiency of modern railway infrastructure, research is shifting toward condition-based maintenance systems that continuously collect and analyze track deviation data, such as gauge, cant, and rail alignment. Kim et al. [24] emphasized the paradigm shift in domestic railway track maintenance based on Fourth Industrial Revolution technologies, raising the necessity for intelligent data management. Furthermore, Park et al. [25] demonstrated a framework for reverse-engineering geometric parameters for railway maintenance from 3D point cloud data of ballasted track sites, establishing a technical foundation for securing quantitative data. Elberink et al. [26] presented a methodology to automatically estimate rail geometric parameters from MLS data to reconstruct 3D track models. This approach can be utilized as a core technology for reverse-engineering-based track BIM data construction. Additionally, to improve coordinate precision and data integration efficiency, Mahmood et al. [7] presented an integrated model with enhanced defect localization performance to address the lack of standardization in conventional Scan to BIM frameworks. Along with these studies, Gabara et al. [27] proposed a methodology to extract core track parameters with high precision from photogrammetry-based point cloud data, providing a quantitative basis for the reverse-engineering and parameterization of inspection data.

To integrate these point cloud-based reverse-engineering technologies with actual maintenance systems, ongoing studies are aiming to map extracted parameters as attribute information within the BIM environment. Within this trend, recent research moves beyond fragmentary information mapping toward upgrading systems to organically integrate and analyze inspection data within standardized platforms. In this regard, Ciccone et al. [28] demonstrated open BIM applications for conventional railway infrastructure maintenance, emphasizing the critical role that standardized data schemas like IFC play in information structuring and condition-based maintenance decision-making. Specifically, Liu et al. [29] proposed a decision-making framework that comprehensively supports track condition diagnosis, long-term degradation prediction, and maintenance planning optimization through open BIM standard schema expansion, presenting a research direction to extend quantitative reverse-engineering data into practical preventive maintenance systems.

2.2. LLM Agent and API Control Methodologies

As demonstrated in the study by Gao et al. [30] regarding complex data processing in the construction and civil engineering sectors, the adoption of LLM-based technologies is currently proliferating within the construction industry. Specifically, Zheng et al. [31] proposed a framework that integrates LLMs into AEC (Architecture, Engineering, and Construction) automation workflows to efficiently extract complex BIM attribute information through natural language queries. Moving a step further, Jiang et al. [32] proposed a framework that automatically generates architectural models compliant with site information and legal regulations by linking LLMs with C#- and Python-based BIM environments (Autodesk Revit 2024), proving the potential to streamline design workflow decision-making through standardized JSON schemas.

Recent technological trends are rapidly evolving beyond simple information retrieval and code generation toward Agentic AI systems where models independently assess engineering situations and call internal APIs of software. Xu et al. [33] proposed a framework that automates engineering design pipelines by combining multi-modality-based LLM agents. Furthermore, they specifically presented the direction and necessity for upgrading to Agentic workflows where models autonomously coordinate decision-making

beyond predefined rules. For concrete autonomy, Jang et al. [34] developed a framework that autonomously designs and generates detailed layers of complex exterior wall objects through a conversational interface, proving the feasibility of collaborative modeling between users and LLMs. Additionally, Dong et al. [35] successfully implemented an AI BIM coordinator system where five specialized multi-LLM agents collaborate to autonomously author and directly execute BIM-internal API-operation codes based on complex natural language commands from users.

However, despite the advancement of these agent-based autonomous-control studies, the approach where an external LLM controls internal APIs in real time within a professional engineering software environment faces technical limitations. Specifically, it leads to system bloating and reduced scalability. For instance, the study by Dong et al. [35] performed complex API control utilizing a nonstandard messaging pipeline dependent on a specific software environment. However, this structure requires building a separate non-standard pipeline, which increases architectural complexity and restricts scalability to other engineering tools due to the use of nonstandard specifications. As pointed out in the study by Zhang et al. [36], current information systems in the AEC sector suffer from low API availability and a lack of standard protocols for model-level interaction due to the proprietary data schemas and closed ecosystems of major platforms. This technical gap, characterized by the absence of a real-time bidirectional data flow, acts as a critical constraint in environments that require controlling and bidirectionally synchronizing complex rail-alignment data, such as railway infrastructure. Furthermore, to control LLM hallucinations during complex control operations and guarantee precise API execution, an expansion into an autonomous decision-making framework that fuses RAG and MCP is essential.

Table 1 presents a comparative analysis demonstrating how the framework proposed in this study overcomes the limitations of these conventional studies and advances the current state of technology. As presented in Table 1, this study overcomes the conventional limitations of static mapping in terms of Building Information Modeling data integration through MCP-based real-time bidirectional data synchronization. In addition, from the perspective of API control and scalability, this study maximizes portability between heterogeneous platforms by introducing a standard protocol instead of a nonstandard pipeline dependent on specific software. Finally, in the area of decision-making autonomy, conventional agent systems faced the risk of hallucinations and control errors due to their reliance on specific rules or autonomous code generation. This study resolves this limitation through the fusion of a RAG-based search of railway maintenance guidelines and a real-time MCP tool environment. Consequently, this framework makes a unique contribution by minimizing the hallucination phenomenon and realizing high-reliability decision-making.

Table 1. Comparison of conventional and proposed methods.

Category	Conventional Approaches	Limitations	Proposed Method	Contribution
BIM Data Integration [25–29]	- Geometric parameter extraction - Standard schema extension & one-way mapping	- Lack of real time - Bidirectional synchronization	MCP-based bidirectional data mapping	- Overcomes one-way mapping limits - Secures organic bidirectional data exchange

API Control & Scalability [35,36]	- Proprietary, non-standard pipelines dependent on specific BIM software - Low API availability - Loss of interoperability with other engineering tools	Introduction of MCP architecture	- Eliminates platform dependency - Maximizes cross-software connectivity & API execution
Decision-Making Autonomy [31–33,36]	- Fragmentary query-based extraction and rule-based control - Risk of hallucination & contextual limitations during complex control	- RAG-integrated retrieval of railway maintenance guidelines - Fusion of MCP real-time tool & schema context	- Minimizes hallucinations via RAG-driven guideline - Ensures error-free API execution via real-time tool & schema guidance

3. Materials and Methods

3.1. System Architecture

Figure 2 illustrates the comprehensive configuration and data circulation structure of the proposed Agentic BIM framework. This system is designed to integrate track-inspection data, BIM models, and maintenance guidelines into a unified, intelligent loop. To guarantee computational independence at each operational stage from data input to cognitive visualization while enabling real-time synchronization through MCP, the overall architecture utilizes a three-stage core process and four distinct intelligent layers. Specifically, as shown at the top of Figure 2, the three-stage process of the system consists of (1) data input and normalization, (2) automated modeling and transaction processing, and (3) semantic feedback and visualization. This sequence controls the overall data flow. The specific functions and data flows of each layer within the detailed framework are as follows.

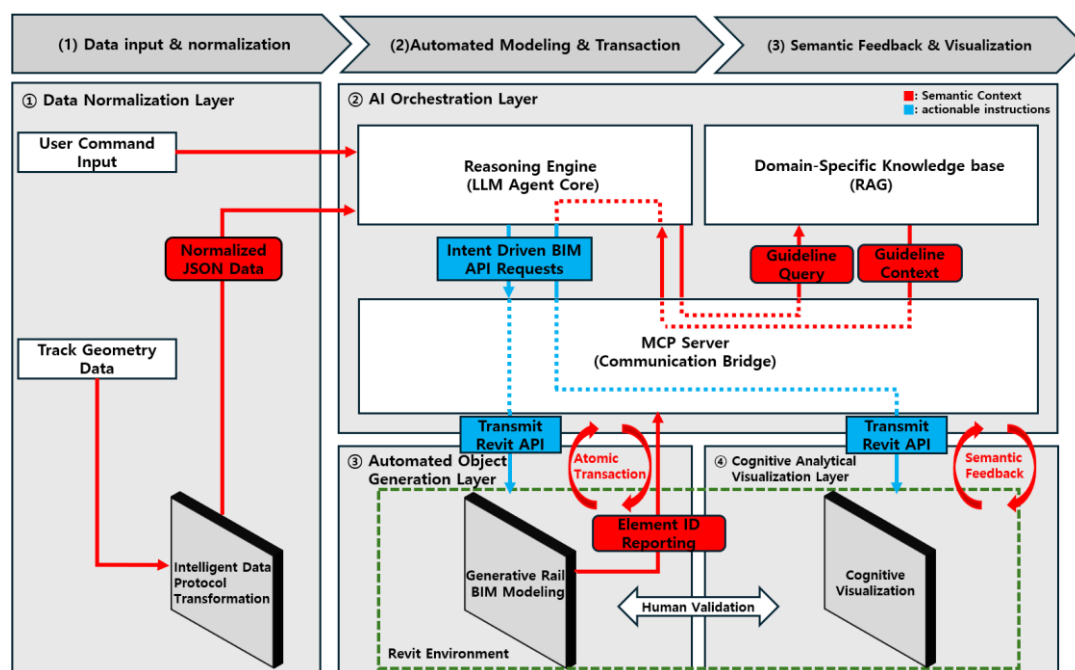


Figure 2. Agentic BIM framework for intelligent rail maintenance.

First, the data normalization layer converts unstructured data collected from external sources and user commands into a specification that the system can process. Natural language commands from users and rail inspection data are transformed into normalized

JSON data through an intelligent data-protocol conversion process. This integration of diverse data into an identical schema secures operational efficiency within the AI orchestration layer.

Second, the AI orchestration layer performs the core role of this framework, combining a reasoning engine and a RAG module. In this study, Claude Desktop is adopted as the reasoning engine. Considering the control characteristics of the Revit API, where minor syntax errors lead to forced program termination, ensuring the overall operational stability of the system is paramount. Therefore, a Claude series model was selected, which has been verified to possess the highest reliability and precision in complex code generation and external tool execution compared to other models [37]. The reasoning engine analyzes the normalized data to generate intent-based tool requests, while concurrently calling specialized knowledge, such as the Railroad Track Maintenance Guidelines, in real time through the RAG module. Notably, to ensure organic synchronization of the system, a MCP implementation strategy was introduced instead of conventional one-to-one custom servers. The MCP strategy of this framework is materialized into three phases: (1) agent server integration via local configuration files, (2) dynamic discovery to perceive tool lists, and (3) secure bridge communication that triggers asynchronous event handlers in Revit. This standard protocol-based architecture guarantees independence between the agent and the control engine. Consequently, it provides high scalability to respond flexibly without modifying the overall communication logic even when new inspection functions or tools are added in the future.

Third, the automated object creation layer is the layer where geometric modeling is performed within the Revit environment. Revit API execution commands transmitted through the MCP server are executed under an atomic transaction management mechanism. This ensures model integrity even during mass data processing. Geometry and attribute information generated in this process pass through a meticulous geometric verification and approval stage by an engineer before being finalized in the database.

Fourth, the cognitive analysis visualization layer is the final stage where the analysis results of the agent are delivered to the user. A graphic override for specific rail sections is executed according to the RAG-based threshold analysis results. Engineers then perform final decision-making by directly reviewing and approving the visualized risk zones. The semantic feedback generated during this process circulates back as an input value for the entire system loop. Consequently, the combination of these four layers provides an autonomous Building Information Modeling operational environment connected in a single pipeline from inspection data analysis to visual decision-making support.

3.2. Intelligent Data Protocol Transformation

To transmit user-defined rail-inspection data attributes to the Revit execution engine (C#) via the AI Agent, a dynamic mechanism capable of converting unstructured data into a structured command hierarchy is essential. This section presents a domain-specific schema design and communication protocol engineered to ensure data integrity and consistency between the agent and the execution engine.

To achieve this, a domain-specific schema predefined as a standardized data format compatible with the object model of the Revit execution engine defines the Rail Data and Highlight Data classes, both tailored to the specific characteristics of railway infrastructure. The Rail Data Class specifies geometric attributes, including X, Y, Z coordinates and rotation values, for the precise physical placement of rail objects. To flexibly accommodate varying inspection attribute names across different field sites, this class incorporates a dictionary-based Parameters structure, thereby ensuring the scalability required to support diverse inspection properties without modifying the schema. The Highlight Data class serves as a dedicated model for visualizing objects that exceed specific thresholds based

on the RAG analysis results. By containing the unique identification numbers of the target objects and their corresponding color values (R, G, B), it functions as an interface that directly maps status grades into visual attributes within the Revit software.

Based on the designed schema, the Agent analyzes user intent and converts it into a JSON payload that can be immediately processed by the API Controller of the Revit execution engine. When the Agent packages the static information of the rail inspection data files and real-time inference results into the Rail Data or Highlight Data format, unstructured attribute names specified by the user are mapped as key value pairs within the Parameters dictionary and serialized. The finalized JSON data is transmitted via the HTTP protocol to the `api/elements/create` or `api/elements/highlight` endpoints of the Revit execution engine. This process implies that the inference results from the AI are directly injected into the Revit execution logic without separate intermediate processing or manual mapping. Through this series of intelligent data-protocol conversion mechanisms, the autonomous automation of the pipeline extending from natural language commands to actual model generation and visual control is achieved.

3.3. Generative Rail BIM and BIM Object Recognition and Synchronization

This section presents the mechanism by which the Revit execution engine controls the Revit API based on the standardized JSON data from Section 3.2 to dynamically generate rail BIM models and synchronize the status of the generated objects with the AI Agent. The received inspection data undergoes a precise preprocessing sequence through a Python-based analysis module connected to the LLM.

As illustrated in Figure 3, the system initiates Logic 1 by utilizing the `las.py` library to invoke the raw data file, directly parsing individual point coordinates along with the attribute information defined within the file. Because the extracted raw data contains significantly large coordinate values, it can induce computational errors or trigger instability within Revit. To prevent this, Logic 2 establishes a specific point within the overall data as a reference baseline and performs a coordinate translation across all data points relative to this baseline. This translation normalizes the coordinates, thereby securing computational precision and enhancing the processing efficiency of the Agent. In Logic 3, the system connects the scattered point cloud data to generate a continuous alignment trajectory that smoothly reconstructs the actual form of the railway rail. In the final logic stage four, the optimized geometric trajectory information and maintenance attribute data are integrated into a single data package. This information is packaged into a JSON specification to enable the agent to execute commands through the MCP server. If preprocessing is not performed properly during this process, consistency errors will occur in the subsequent rail model creation stage. To mitigate this risk, an exception handling system was implemented to prevent object creation under such errors. Consequently, this design allows end users to intuitively verify the stability of the preprocessed data and the readiness of the subsequent steps based on the successful generation of the model.

Once this structured data is delivered to the Revit execution engine, the system activates an event-driven command-execution architecture tailored to the single-threaded operational nature of Revit. Generation commands received from an external agent via the HTTP server are forwarded to the Revit main process through `ExternalEvent.Raise()`. This schedules the execution of API commands by invoking the event handler when Revit is in an idle state, serving as a core mechanism to prevent system crashes and ensure communication stability. Furthermore, the entire modeling process is executed within an atomic transaction object. This safeguards model data integrity by maintaining database consistency and triggering a rollback if an error occurs, even when the Agent transmits massive volumes of rail inspection data.

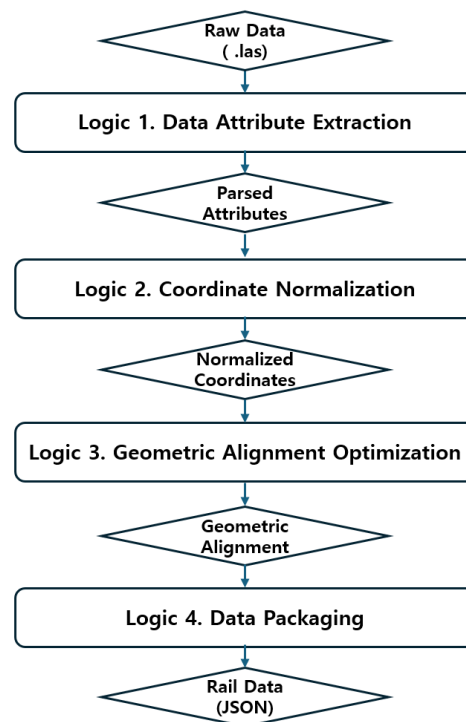


Figure 3. Preprocessing logic for inspection data.

In the final stage of the modeling process, raw data attributes are injected into the generated objects, and the execution results are reported back to the Agent to establish intelligent control. The user-defined attributes contained in the parameter dictionary of the rail object are directly implanted into Revit instance parameters through the LookupParameter method, creating an engineering information model that goes beyond simple geometric shapes. Immediately following the modeling sequence, the Element ID of the created object is processed into the Element Info class format and transmitted back to the Agent, forming a feedback loop. Through this cognitive loop, the AI Agent identifies the actual identifiers of the generated objects in real time, serving as a foundational basis to precisely identify and synchronize targets when a user subsequently requests specific ID-based visualization or status analysis. Consequently, this series of processes generates objects where physical attributes such as Gauge, Cant, Align, and Tp are directly injected as parameter values, establishing a BIM environment with secured data consistency.

3.4. Cognitive Visualization and Analysis

Based on the object recognition and synchronization framework established in Section 3.3, this section describes a visualization mechanism for intuitively projecting the engineering judgment results derived by the AI Agent into the Revit environment. This system is designed so that the analysis results of the agent do not merely remain as simple text reports. Instead, it provides immediate cognitive feedback to managers through graphic control on the BIM model. The AI Agent changes the visual attributes of specific objects in real time based on the error analysis or status diagnosis results of the rail inspection data. Figure 4 shows the internal logic of the system where the extracted actual inspection values are compared and analyzed in real time with the management threshold values in the track maintenance guidelines.

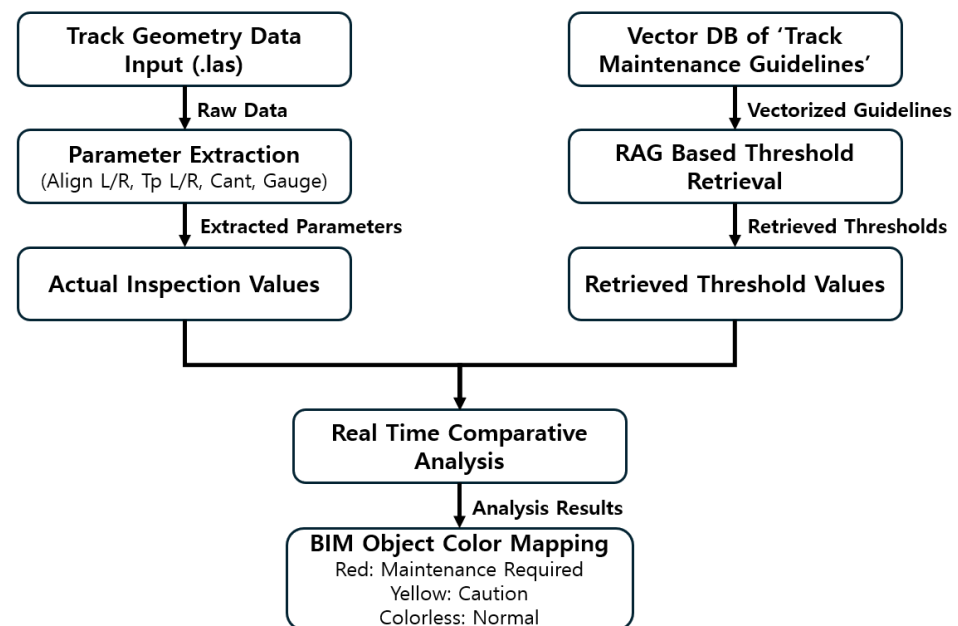


Figure 4. RAG-based threshold analysis and classification logic.

The introduction of a RAG architecture integrated with a Vector Database instead of LLM fine-tuning when determining track maintenance standards in this study is to guarantee engineering safety and reliability. Because management threshold values defined in railway safety standards and regulations require high precision, LLM hallucination errors, which are a concern during fine-tuning, can lead directly to accidents. The RAG module verifies the rigor of the Agent judgment and compliance with regulations by referencing and mapping the threshold values of actual guidelines as a static source. Furthermore, it secures the traceability of the decision-making path.

Based on this intelligent loop, the agent classifies the input inspection errors into “Maintenance”, “Caution”, and “Normal” grades by contrasting them with the RAG guideline standards. When executing a command to identify sections requiring maintenance that exceed the thresholds, the agent maps the Element ID of the object previously secured in Section 3.3. Concurrently, it determines the RGB color values corresponding to each grade, such as Red for maintenance, Yellow for caution, and Green for normal.

The determined color information and target identification data are assembled into a JSON packet in accordance with the previously defined Highlight Data class specification. This process signifies the conversion of the qualitative judgment of the AI into a quantitative instruction that the Revit execution engine can understand. Subsequently, the assembled data is transmitted to the endpoint of the Revit execution engine via the HTTP protocol.

The Highlight Elements Event Handler of the Revit execution engine controls the graphic settings of the Revit API based on the received JSON data to perform real-time visualization. To secure visibility within Revit, it first utilizes the Filtered Element Collector to search for the Fill Pattern Element within the project and identifies and activates a solid color pattern. Subsequently, by calling the Set Projection Line Color and Set Surface Foreground Pattern Color methods for specific objects, it changes the line and surface colors of the corresponding object in the currently active view. At this time, instead of permanently modifying the physical attributes of the original family, it performs a real-time override at a specific view level. This method provides the flexibility to visualize data analysis results while maintaining the integrity of the original model data. Consequently, without the need to passively interpret voluminous reports, users can intuitively identify

maintenance priority locations through the highlighted model, thereby establishing rapid and objective maintenance decision-making.

4. Result

4.1. LLM-MCP Server Implementation

In this study, a server environment was established to verify the effectiveness of the proposed LLM-BIM integration system. The server was configured based on the organic linkage of Autodesk Revit software (2024), a Large Language Model (Claude 4.3 Desktop), an MCP server, and a RAG module utilizing track maintenance guidelines data.

The reason for adopting the Autodesk Revit environment as the final verification engine instead of open source models based on the open standard IFC at this stage is that it considers practical effectiveness and the actual status of technology dissemination in the AEC industry. Among the representative BIM software utilized by industry professionals, Revit demonstrates the highest market share at approximately 76.5% [38]. Therefore, in proving the practical applicability of the proposed Agentic BIM framework, utilizing the most widely disseminated Revit platform is most suitable for verifying substantial effects. Consequently, this study established the verification environment based on the Revit API to secure this practical alignment.

In the MCP server operation architecture that plays a pivotal role in the system, user natural language commands are classified and processed into two core workflows. First, the Revit MCP Python Server receives automated modeling commands to create rail objects within the Revit environment based on track inspection data. Second, the Langconnect RAG MCP Server provides visual information by highlighting specific sections of the model by color based on maintenance information extracted from the RAG module according to visualization commands. As a result of monitoring, it was confirmed that each MCP server node was independently activated, enabling data exchange and command transmission between the LLM environment and the Revit execution engine to be performed without delay. This proved that the dualized mediation structure based on MCP in the LLM-BIM integrated system operates in accordance with the design intent. Consequently, a stable server-operating environment for real-time control and data synchronization between heterogeneous platforms was secured.

4.2. Three-Stage Intelligent Interaction

This section evaluates the interaction among the user-LLM-BIM via the implemented MCP server, with testing structured into two subsequent phases: "Generation and Feedback" and "Cognitive Visualization". To isolate the basic operational performance of the system from the confounding variables of geometric complexity, the evaluation was conducted specifically on a straight section of a railway track.

4.2.1. Automated Rail Modeling and ID Feedback Loop

In the validation experiment of this study, the geometric properties of the railway rail (Gauge, Cant, Align, Tp, etc.) were defined as input parameters, and the system was designed so that the LLM engineeringly interprets this data to automatically generate 3D rail models within the Revit software environment. The modeling commands from the LLM are forwarded to the Revit API through the MCP server. Inside Revit, rail objects are dynamically generated according to the received parameters, and a unique ID is assigned to each object. To secure data persistence so that the generated object information can be referenced in subsequent processes such as user queries, filtering, and RAG-based visualization, the data is synchronized with the MCP database.

Specifically, upon receiving an unstructured natural language command such as “generate rails based on rail inspection data,” the proposed framework goes beyond simple text-matching to semantically analyze the modeling intent of the user and the required geometric constraints. In this process, the system sequentially controls a data reference stage to identify the structure of external inspection data, a numerical optimization stage to process the extracted inspection data for BIM software integration, and an instantiation stage to construct the geometry by invoking the Revit API based on the processed data. The Agent acts as an orchestrator that binds these engineering procedures into a single operational workflow, executing a complex engineering pipeline.

Data analyzed through the process illustrated in Figure 3 is successfully synchronized in real time with the Revit API via the MCP, moving beyond the generation of simple geometric shapes to automatically construct 3D rail objects containing comprehensive attribute information. As shown in Figure 5, this evaluation confirmed that essential physical properties required during the maintenance phase including Gauge, Cant, Align, and Tp are directly embedded as parameter values into the generated rail objects, thereby ensuring data consistency.

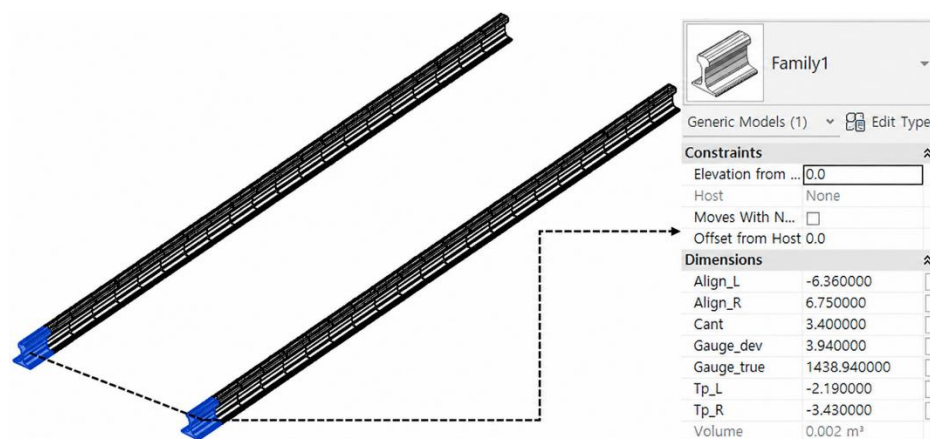


Figure 5. Automated rail modeling and attribute injection results.

This automated generation process guarantees data alignment between the inspection data and the BIM model, while establishing a database mediated by unique IDs to provide the structural foundation for the subsequent RAG-based precision visualization and condition analysis.

4.2.2. Object Highlighting and Decision Support

Utilizing the data alignment secured through the MCP, the proposed framework establishes an intelligent visualization system that links field inspection data with maintenance guidelines. The integrated Agent utilizes the RAG architecture to reference prescribed thresholds in real time, autonomously deducing the deviation status and maintenance requirements of the generated BIM objects. By visually mapping the derived condition information directly onto the model, this process transforms the conventional static geometry into an active information model that directly supports administrator decision-making.

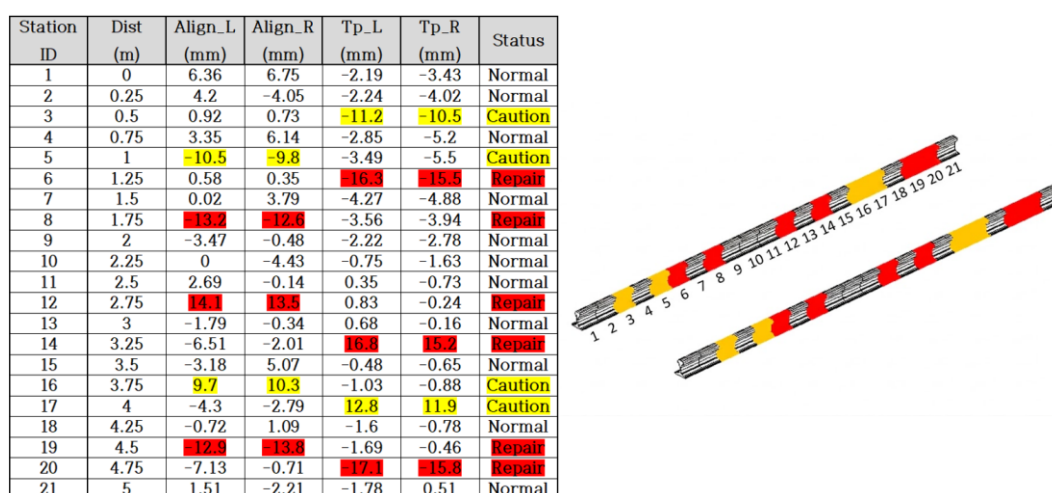
The automatically generated rail objects retain unique IDs and attribute parameters synchronized with the MCP database. The Agent executes the RAG module to invoke the “railway track maintenance guidelines” [23] stored in the Vector Database. Specifically, to validate the precision of the system, this study adopted the general railway maintenance standards for sections with operating speeds of 80 to 120 km/h as the criteria, and Table 2 presents the detailed thresholds for each grade applied in the experiment.

Table 2. Maintenance standards and irregularity thresholds for conventional railway.

Maintenance Standards for Conventional Railways ($80 \leq V \leq 120$ km/h)			
	Longitudinal Level Irregularity (Tp) (mm)	Alignment Irregularity (Align) (mm)	Gauge Irregularity (mm)
Caution	≥ 10 mm	≥ 9 mm	< -3 mm or ≥ 17 mm
Repair	≥ 15 mm	≥ 12 mm	< -5 mm or ≥ 20 mm

The Raw Data collected from the test track consists of precise actual measurement data obtained from an operating railway line. However, due to the thorough preventive maintenance and safety management system of the domestic railway, all collected measurement data remained within a highly stable state below the safety management thresholds. Consequently, a limitation of data imbalance existed, where actual defect data was scarce for multifaceted verification of the system classification and visualization logic. Therefore, to clearly verify the effectiveness of the “Maintenance” and “Caution” grade identification capabilities and highlighting functions of this framework, virtual defect scenarios were constructed for testing. This was achieved by scaling the numerical ranges of certain inspection parameters to precisely reflect the “Caution” and “Maintenance” threshold criteria in the relevant railway maintenance guidelines, while completely preserving the physical characteristics of the acquired measurement data.

The classified condition grades are instantly rendered on the model through the Revit API Highlight Elements Tool. The system executes a visualization process that enables engineers to intuitively identify maintenance priorities among multiple rail objects. As shown in Figure 6, rail segments classified under the maintenance grade are highlighted in red to immediately deliver a visual warning. Moving beyond simple color emphasis, the system links the recommended maintenance deadlines extracted from the guidelines by the Agent with the object information, simultaneously displaying them as embedded text annotations within the viewport. This provides an environment where administrators can establish immediate maintenance scheduling directly within the model, eliminating the need to cross-reference separate documents. In contrast, caution sections which have reached the threshold for proactive management while remaining within ultimate safety limits are visualized in yellow. This serves as a key indicator supporting preventative-maintenance-centered decision-making by preeminently visualizing potential defect points within large-scale railway infrastructure.

**Figure 6.** Track irregularity classification and rail BIM visualization results.

Consequently, this process moves beyond static modeling that reproduces simple geometric shapes, realizing an intelligent information model where field inspection data and regulatory management standards are synchronized within a single semantic model. While this approach does not achieve full synchronization based on real-time sensor streaming, it holds significant engineering value by laying a critical steppingstone toward a future autonomous maintenance-based railway digital twin, in that it diagnoses physical conditions and materializes decision-making through the autonomous reasoning of the Agent.

When a user selects specific highlighted rail objects (ID: 12 and 14), the automatically embedded precision attribute data and the grade status information determined by the Agent are output together as shown in Figure 7, thereby ensuring objective decision-making. Furthermore, rails classified under the maintenance grade display the specific repair deadlines stipulated in the track-maintenance guidelines, providing clear priority criteria for establishing rail-maintenance schedules. This mechanism that fuses textual maintenance guidelines with visual object attributes significantly alleviates the cognitive load of conventional maintenance workflows, which previously required individual interpretation of massive amounts of numerical data.

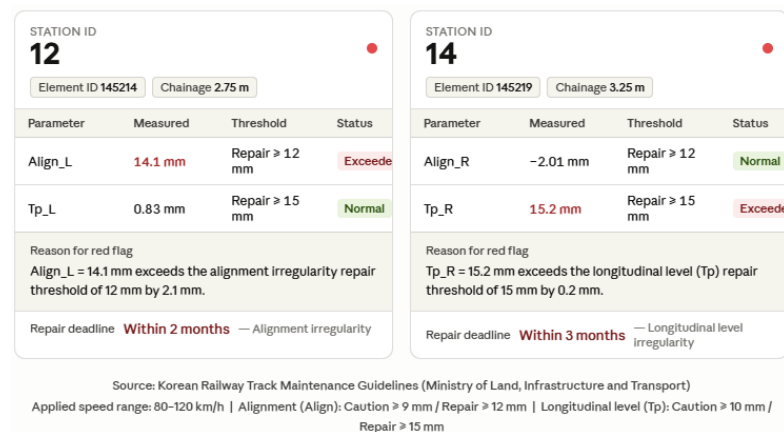


Figure 7. Rail maintenance diagnosis results for Station ID 12 and 14.

This data knowledge coupled visualization framework minimizes the time required from defect detection to maintenance planning, thereby securing the timeliness of railway infrastructure maintenance and establishing the technical foundation to preemptively prevent safety accidents caused by human error.

4.3. Performance Analysis

A quantitative performance evaluation was performed to verify the engineering effectiveness and reliability of the proposed Agentic BIM framework in this study. In this evaluation, to prove the engineering validity of the system, an analysis was conducted based on three core engineering indicators: hallucination suppression performance, execution and response latency, and object-creation precision. To achieve this, a comparative analysis was performed using the traditional manual analysis workflow and the framework proposed in this study as control groups. In particular, the experimental conditions were subdivided and verified according to whether the internal RAG system of the proposed framework was integrated.

To first verify the decision-making hallucination suppression performance of the proposed framework, a maintenance grade determination analysis according to RAG integration was performed using the inspection data constructed in Section 4.2.2. In this experiment, a total of thirty independent experiments were repeatedly performed in the

same environment to secure the reproducibility and reliability of the results. Additionally, to equally evaluate the determination performance of each grade and verify the overall classification balance, an arithmetic mean method, the Macro-average indicator, was applied. The results are shown in Table 3.

Table 3. Performance comparison according to RAG integration.

	Accuracy	Precision	Recall	F1-Score	Hallucination Rate
LLM Only	72.54 ± 16.14%	70.82 ± 18.00%	67.28 ± 16.71%	68.80 ± 17.06%	27.78 ± 16.14%
LLM + RAG	100%	100%	100%	100%	0%

Note: All values represent the mean ± SD (standard deviation) of 30 replicated trials.

According to the experimental results, the overall decision-making accuracy in the environment where decision-making was performed relying solely on the LLM without integrating the RAG module remained at the level of 72.54 ± 16.14%. In particular, the hallucination rate where the model failed to discriminate precise numerical threshold conditions and assigned grades different from the ground truth reached 27.78 ± 16.14%, confirming high reasoning errors and performance deviations. This high standard deviation occurring without RAG shows that the variability of judgment is maximized by the probabilistic generation mechanism inside the standalone LLM even under the same input data and conditions. Analyzing the detailed misclassification tendencies, not only over-detection misjudging actual normal data as a caution grade but also errors classifying defects in maintenance and caution stages as normal occurred frequently in both directions. Consequently, the overall precision representing the classification reliability of the model deteriorated to 70.82 ± 18.00%, the overall recall to 67.28 ± 16.71%, and the overall F1-Score to 68.80 ± 17.06%. This phenomenon occurs because the model lacking domain-specific knowledge fails to accurately identify the quantitative thresholds of the track structure. When applied in practice, it becomes a direct factor that causes unnecessary administrative costs or undermines the safe train operation and normal track management.

On the other hand, in the environment where the exact inspection guidelines were injected in real time through the RAG module, the entire test data perfectly matched the ground truth in all thirty repeated experiments. Consequently, the system achieved 100% in all indicators including accuracy, precision, recall, and F1-Score, while the hallucination rate was completely suppressed to 0%. The reason why the proposed framework could demonstrate such high accuracy and stability is due to the structural characteristics of the maintenance grade determination task performed by this system. This task is not a free generation task that interprets ambiguous contextual backgrounds. Instead, it possesses the nature of a deterministic mapping task that classifies data 1:1 according to standardized guideline numerical standards. Therefore, when explicit quantitative threshold conditions are seamlessly secured within the prompt through RAG, the model does not rely on internal probabilistic reasoning or virtual information generation. Instead, it performs the determination by substituting the delivered reference rules into the inspection data. In other words, this proves that the RAG system goes beyond simply supplementing external knowledge, functioning as a core mechanism to control uncertain numerical calculation errors and hallucination phenomena inside the LLM and to secure the safety and reliability of the system.

Second, to verify the execution speed and work efficiency of the proposed framework, the elapsed time from the point of user-natural language command input to the stage where automated object creation and visualization within Revit were completed was measured. The test was based on the rail object modeling and inspection data mapping work of a 10 m straight section. The average elapsed time was calculated through a total

of 10 independent simulations. Table 4 and Figure 8 quantitatively present the detailed elapsed time for each stage and the comprehensive efficiency comparison results.

Table 4. Quantitative comparison of execution time and operational efficiency by stage.

Workflow Stage	Task Description	Traditional Workflow(min)	Agentic BIM Framework (min)	Time Reduction (%)
Stage 1	Data Packaging & Preprocessing	N/A	3.1 ± 0.3	N/A
Stage 2	Object Creation	55.0 ± 5	2.0 ± 0.2	96.4%
Stage 3	Color Visualization Mapping	30.0 ± 3	5.1 ± 0.4	83.0%
Total	Overall End-to-End Elapsed Time	88.5 ± 8	10.2 ± 0.6	88.5%

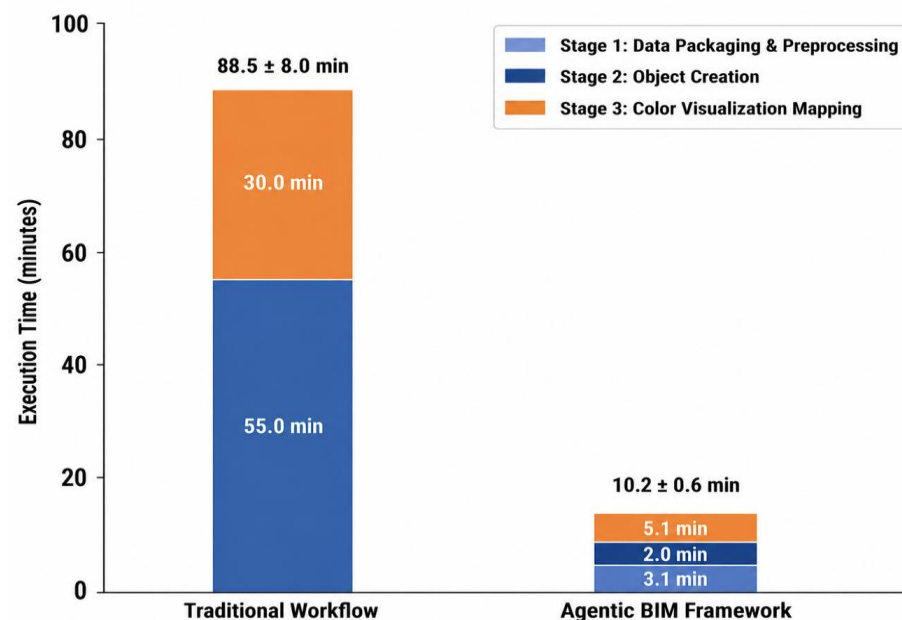


Figure 8. Execution time reduction by stage.

As shown in Table 4, as a result of analyzing the traditional manual workflow, it took approximately 55.0 ± 5 min for an engineer to create 3D rail objects while manually contrasting inspection data and input-precise attribute information. In addition, approximately 30.0 ± 3 min were added to the task of classifying into each risk grade by contrasting the track maintenance guidelines and changing object colors within Revit for visualization. Consequently, an average work time of 88.5 ± 8 min was required to complete the entire manual process. As can be visually confirmed in Figure 8, in the manual method, the object creation and attribute input stage (Stage 2) appeared as a bottleneck point occupying more than half of the total time. This possesses a limitation where the required time and human error probability increase exponentially as the scale of data to be processed expands.

On the other hand, when applying the Agentic BIM framework proposed in this study, the total time required to complete the same scenario appeared as 10.2 ± 0.6 min, demonstrating an overall elapsed time reduction effect of approximately 88.5% compared to the traditional manual method. As shown in Table 4, the data-packaging stage (Stage 1) that performs automated preprocessing and standardization of raw inspection data

required 3.1 ± 0.3 min, contributing to early framework stabilization. In particular, the stage of automatically calling the Revit API based on the MCP module to create 3D objects (Stage 2) was shortened from 55 min in the manual method to an average of only approximately 2 min by approximately 96.4%, showing the highest efficiency improvement within the framework. Subsequently, the stage of mapping RAG analysis results to visualize them in colors by risk grade (Stage 3) took an average of approximately 5.1 min. Although this was the stage that took the longest time within the proposed system, it recorded a reduction rate of 83.0% compared to the manual method, demonstrating that rapid model reflection of inspection data and efficient visual decision-making support are possible. Beyond simple speed improvements, these results prove high execution stability during practical application through a small standard deviation. Furthermore, by blocking data mapping errors that may occur during manual work while shortening the model reflection speed, it clearly proves that rapid decision-making based on inspection data and immediate visualization feedback can be stably provided in a practical environment.

5. Discussion

This study proposes a framework in which an LLM Agent autonomously controls BIM software through the medium of an MCP server and a RAG architecture, and verifies the automation of the entire process. This achieves a methodological advancement that expands the Scan-to-BIM paradigm from quantitative mapping to the automation of engineering judgment. This is accomplished by implementing a decision-making system that interprets the context of maintenance guidelines and judges complex conditions.

Mahmood and Kim's work [7], a study attempting to extract geometric parameters from traditional point cloud data and integrate them with BIM, presented a framework for extracting indicators and mapping them to an IFC 4.3-based BIM model. However, because it focused on single threshold-based defect identification, it possessed limitations in expanding into comprehensive decision-making that links complex aspects with voluminous manuals. In contrast, the Agent of this study infers the context of guidelines through RAG to interpret complex situations on its own and flexibly responds to regulatory changes. Additionally, Sampaio et al. [15] demonstrated efficiency by linking external data with BIM using Dynamo and Python. Nevertheless, their approach exhibited rigidity, requiring operator intervention for object creation and relying on procedural scripts that necessitate modifying the entire code when regulations change. This framework compensates for such limitations through natural-language command-based API control. Consequently, it holds engineering significance in alleviating rigidity and presenting a highly scalable maintenance workflow.

Furthermore, when reviewing the results of this study from the perspective of a large-scale railroad network, technical limitations that must be overcome for actual field system deployment and practical application are derived. First, there is the issue of computational bottlenecks and execution delays during large-capacity data-processing. When processing large-scale, high-density inspection data on actual railroad lines spanning hundreds of kilometers, the real-time embedding computation of the RAG pipeline and the sequential inference loop between agents aggravate the system overload. Because this causes delays in decision-making, research to selectively search and update in real time only specific segment areas that exceed management thresholds or line alignment changes is essentially required, instead of recalculating the entire data every time. Second, verification of inference robustness is insufficient in the severe data imbalance environment peculiar to railroad infrastructure. Due to the preventive maintenance system in the railroad field, it is difficult to collect actual defect data from operating lines. In this study, to demonstrate the discrimination power of the grade-determination logic, a virtual defect scenario scaled according to guideline thresholds while maintaining the physical

characteristics of the actually measured data was applied. However, the engineering judgment stability of the agent under complex sensor noise or multiple interference data conditions in the actual field was not verified. In the future, securing noise datasets from various environments and advancing a control algorithm that verifies the agent control command sequence before execution must be accompanied. Third, limitations in infrastructure deployment exist due to the closed network security constraints of actual railroad systems. The railroad infrastructure environment, which is a national critical infrastructure, is operated in the form of a closed network completely disconnected from the external internet due to security regulations. Therefore, the current operation method relying on commercial external API dependencies cannot satisfy practical standards. Constructing and optimizing an On-Premises domain-specific LLM that can be independently mounted inside the server infrastructure so that the system can operate completely independently without external network linkage is a task that must preemptively precede.

6. Conclusions

6.1. Research Summary and Achievement

The Agentic framework proposed in this study integrates LLM, MCP, and RAG technologies, thereby successfully addressing the inherent challenges of point cloud data pre-processing and object generation, which previously involved excessive manual labor and the potential for information loss or errors during data conversion. The core of this research lies in converting rail inspection data into structured geometric parameters and synchronizing them in real time with the Revit execution environment through an MCP server to generate 3D rail models. By leveraging RAG, the Agent independently navigates track maintenance guidelines and cross-analyzes thresholds, moving beyond simple shape generation to realize cognitive visualization based on inspection results. Evolving into an active decision-making platform that independently perceives and judges the field, this framework is expected to serve as a pivotal foundation for establishing a safe and efficient next-generation railway-operating environment by realizing data-driven proactive maintenance in the future.

6.2. Practical Benefits and Contributions

The Agentic BIM framework proposed in this study maximizes efficiency in railway maintenance practices while presenting a new paradigm for intelligent railway infrastructure management technologies. From a practical standpoint, this framework offers two core benefits. First, through LLM-based orchestration that controls the model via natural language commands without complex API programming, it virtually eliminates quality variances stemming from engineer proficiency and improves task execution speed. Second, by integrating a color-based visual warning system into the automation loop, it enables operators to instantly grasp maintenance priorities and expedites decision-making through highlighted objects, eliminating the need to cross-reference extensive numerical reports. These practical achievements serve as a critical stepping stone, evolving information models which were previously confined to post hoc records or static 3D visualization tools into practical autonomous decision-making support systems. By automating the sequential process where the AI Agent perceives field data, assesses conditions according to maintenance regulations, and directly controls the BIM execution environment, this study is significant in that it successfully demonstrates the concept of an Agentic autonomous operating system where judgment and execution are fully unified.

6.3. Limitations and Future Research Directions

This study successfully demonstrated the automation workflow possibility of the Agentic BIM framework targeting ballast straight tracks. However, for this system to fully settle as an intelligent autonomous maintenance system in actual field operations, strict limitations and management risks exist as follows. First, there are limitations in spatial scope expansion and geometric optimization. To precisely process cant- and gauge-widening data of curved and turnout sections in accordance with actual large-scale railroad networks, upgrading the agent inference logic is essential. Concurrently, optimizing algorithm-based parametric generation logic that can overcome the dynamic alignment control constraints of the Revit API is required. Along with this, cyber security standards and closed-network deployment constraints required by the nature of national infrastructure exist. Furthermore, challenges remain regarding inference uncertainty due to the non-deterministic characteristics of the LLM and securing model reliability. Consequently, beyond the virtual scenario verification stage, reproducibility in various actual track inspection dataset environments and empirical verification through large-scale railroad networks have not been sufficiently conducted. In conclusion, until these security and technical vulnerabilities, inference uncertainty, and multifaceted verification based on large-scale lines are completely resolved at the production level, the immediate applicability in practice must be viewed as limited. In the future, the advancement of the intelligent maintenance framework will be promoted through research on spatial expansion including curves and turnouts, upgrading a control algorithm that pre-verifies agent control commands, and establishing an update system that perfectly synchronizes the static data environment with real-time dynamic field data.

Author Contributions: Literature review, M.J.; writing—original draft preparation, M.J.; proposal of the overall framework and direction of the research, S.K.; writing—review and editing, S.K.; dataset generation M.J. and Y.K.; data analysis, M.J.; data analysis review, Y.K. and S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (RS-2023-NR076575). Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agency.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: The authors also express their gratitude to the research team members for their support throughout this study.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Rempelos, G.; Ognibene, G.; Le Pen, L.; Blainey, S.; Preston, J.; Powrie, W. Railway Track Deterioration Models: A Review of the State of the Art. *Transp. Geotech.* **2024**, *49*, 101377. <https://doi.org/10.1016/j.trgeo.2024.101377>.
2. Fischer, S.; Németh, A.; Horvath, H. Investigation of the track gauge in curved sections, considering Hungarian railway lines. *Infrastructures* **2023**, *8*, 69. <https://doi.org/10.3390/infrastructures8040069>.
3. Liao, Y.; Han, L.; Wang, H.; Zhang, H. Prediction Models for Railway Track Geometry Degradation Using Machine Learning Methods: A Review. *Sensors* **2022**, *22*, 7275. <https://doi.org/10.3390/s22197275>.
4. Hwang, S.H.; Kim, S.; Lee, K.C.; Jang, S.Y. Effects of Long-Wavelength Track Irregularities Due to Thermal Deformations of Railway Bridge on Dynamic Response of Running Train. *Appl. Sci.* **2018**, *8*, 2549. <https://doi.org/10.3390/app8122549>.
5. Guler, H. Decision Support System for Railway Track Maintenance and Renewal Management. *J. Comput. Civ. Eng.* **2013**, *27*, 292–306. [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000221](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000221).

6. Xie, J.; Huang, J.; Zeng, C.; Jiang, S.H.; Podlich, N. Systematic Literature Review on Data-Driven Models for Predictive Maintenance of Railway Track: Implications in Geotechnical Engineering. *Geosciences* **2020**, *10*, 425. <https://doi.org/10.3390/geosciences10110425>.
7. Mahmood, B.; Kim, S. Framework of Scan to Building Information Modeling for Geometric Defect Localization in Railway Track Maintenance. *Buildings* **2024**, *14*, 3578. <https://doi.org/10.3390/buildings14113578>.
8. Maksoud, A.; Alawneh, S.I.A.R.; Hussien, A.; Abdeen, A.; Abdalla, S.B. Computational Design for Multi-Optimized Geometry of Sustainable Flood-Resilient Urban Design Habitats in Indonesia. *Sustainability* **2024**, *16*, 2750. <https://doi.org/10.3390/su16072750>.
9. Costin, A.; Adibfar, A.; Hu, H.; Chen, S. Building Information Modeling (BIM) for transportation infrastructure—Literature review, applications, challenges, and recommendations. *Autom. Constr.* **2018**, *94*, 257–281. <https://doi.org/10.1016/j.autcon.2018.07.001>.
10. Howard, R.; Björk, B.C. Building information modelling—Experts' views on standardisation and industry deployment. *Adv. Eng. Inform.* **2008**, *22*, 271–280. <https://doi.org/10.1016/j.aei.2007.03.001>.
11. Elsayed, S.; Ali, M.; Gupta, D. Orchestrating LLM-Powered Workflows for Autodesk Revit via Model Context Protocol: A Multi-Agent Framework for Intelligent BIM Automation. In Proceedings of the International Conference on Computing in Civil and Building Engineering (ICCCBE), Taipei, Taiwan, 23–26 March 2026.
12. Tang, S.; Shelden, D.R.; Eastman, C.M.; Pishdad-Bozorgi, P.; Gao, X. A review of building information modeling (BIM) and the internet of things (IoT) devices integration: Present status and future trends. *Autom. Constr.* **2019**, *101*, 127–139. <https://doi.org/10.1016/j.autcon.2019.01.020>.
13. Pregnotato, M.; Gunner, S.; Voyagaki, E.; De Risi, R.; Carhart, N.; Gavriel, G.; Tully, P.; Tryfonas, T.; Macdonald, J.; Taylor, C. Towards Civil Engineering 4.0: Concept, workflow and application of Digital Twins for existing infrastructure. *Autom. Constr.* **2022**, *141*, 104421. <https://doi.org/10.1016/j.autcon.2022.104421>.
14. Volk, R.; Stengel, J.; Schultmann, F. Building Information Modeling (BIM) for existing buildings—Literature review and future needs. *Autom. Constr.* **2014**, *38*, 109–127. <https://doi.org/10.1016/j.autcon.2013.10.023>.
15. Sresakoolchai, J.; Kaewunruen, S. Integration of Building Information Modeling and Machine Learning for Railway Defect Localization. *IEEE Access* **2021**, *9*, 166039–166047. <https://doi.org/10.1109/ACCESS.2021.3135451>.
16. Dadashi, N.; Golightly, D.; Sharples, S. Modelling decision-making within rail maintenance control rooms. *Cogn. Technol. Work* **2021**, *23*, 255–271. <https://doi.org/10.1007/s10111-020-00636-x>.
17. Kaewunruen, S.; AbdelHadi, M.; Kongpuang, M.; Pansuk, W.; Remennikov, A.M. Digital Twins for Managing Railway Bridge Maintenance, Resilience, and Climate Change Adaptation. *Sensors* **2023**, *23*, 252. <https://doi.org/10.3390/s23010252>.
18. Ma, W.; Guo, Y.; Han, Q.-L.; Zhou, W.; Zhu, X.; Xiong, J.; Wen, S.; Xiang, Y. Understanding Agentic AI: Algorithms and Infrastructure. *IEEE/CAA J. Autom. Sin.* **2026**, *13*, 776–795. <https://doi.org/10.1109/JAS.2026.125993>.
19. Jelodar, M.B. Generative AI, Large Language Models, and ChatGPT in Construction Education, Training, and Practice. *Buildings* **2025**, *15*, 296. <https://doi.org/10.3390/buildings15060933>.
20. Anthropic. Model Context Protocol (MCP). Available online: <https://modelcontextprotocol.io> (accessed on 15 May 2026).
21. Du, J.; Nousias, S.; Borrmann, B. Text2BIM: Generating Building Models Using a Large Language Model-Based Multiagent Framework. *J. Comput. Civ. Eng.* **2026**, *40*, 04025142. <https://doi.org/10.1061/JCCEE5.CPENG-6386>.
22. He, C.; He, W.; Liu, M.; Leng, S.; Wei, S. Enriched Construction Regulation Inquiry Responses: A Hybrid Search Approach for Large Language Models. *J. Manag. Eng.* **2025**, *41*, 04024083. <https://doi.org/10.1061/JMENEA.MEENG-6444>.
23. Korea Rail Network Authority. *Track Maintenance Guidelines*; Korea Rail Network Authority: Daejeon, Republic of Korea, 2015.
24. Kim, E.; Kim, S. A Literature Review of Domestic Research for Railway Maintenance. *J. Korean Soc. Railw.* **2018**, *21*, 800–814. <https://doi.org/10.7782/JKSR.2018.21.8.800>.
25. Park, S.; Kim, S.; Seo, H. Study on Representative Parameters of Reverse Engineering for Maintenance of Ballasted Tracks. *Appl. Sci.* **2022**, *12*, 5973. <https://doi.org/10.3390/app12125973>.
26. Oude Elberink, S.; Khoshelham, K. Automatic Extraction of Railroad Centerlines from Mobile Laser Scanning Data. *Remote Sens.* **2015**, *7*, 5565–5583. <https://doi.org/10.3390/rs70505565>.
27. Gabara, G.; Sawicki, P. A New Approach for Inspection of Selected Geometric Parameters of a Railway Track Using Image-Based Point Clouds. *Sensors* **2018**, *18*, 791. <https://doi.org/10.3390/s18030791>.
28. Ciccone, A.; Di Stasio, S.; Asprone, D.; Salzano, A.; Nicoletta, M. Application of openBIM for the Management of Existing Railway Infrastructure: Case Study of the Cancellor-Benevento Railway Line. *Sustainability* **2022**, *14*, 2283. <https://doi.org/10.3390/su14042283>.

29. Liu, Z.; Kim, J.I.; Yoo, W.S. Decision support for railway track facility management using OpenBIM. *Autom. Constr.* **2024**, *168*, 105840. <https://doi.org/10.1016/j.autcon.2024.105840>.
30. Gao, Y.; Yiu, T.W.; Shen, X.; Tam, V.W.Y. Large language models in smart construction: A systematic review of implementation strategies, applications and future directions. *Eng. Constr. Archit. Manag.* **2026**, *33*, 159–181. <https://doi.org/10.1108/ECAM-10-2025-1668>.
31. Zheng, J.; Fischer, M. Dynamic prompt-based virtual assistant framework for BIM information search. *Autom. Constr.* **2023**, *155*, 105067. <https://doi.org/10.1016/j.autcon.2023.105067>.
32. Jiang, J.X. Large Language Model-Driven Full Lifecycle Architectural Design: A BIM-Based Workflow and Prototype for Apartment Buildings. *Comput. Civ. Eng.* **2025**, *2025*, 812–819. <https://doi.org/10.1061/9780784486436.086>.
33. Xu, H.R.; Zhang, N.; Yin, Z.Y.; Njock, P.G.A. Multimodal framework integrating multiple large language model agents for intelligent geotechnical design. *Autom. Constr.* **2025**, *176*, 106257. <https://doi.org/10.1016/j.autcon.2025.106257>.
34. Jang, S.; Lee, G.; Oh, J.; Lee, J.; Koo, B. Automated detailing of exterior walls using NADIA: Natural-language-based architectural detailing through interaction with AI. *Adv. Eng. Inform.* **2024**, *61*, 102532. <https://doi.org/10.1016/j.aei.2024.102532>.
35. Dong, Y.; Zhan, Z.; Hu, Y.; Doe, D.M.; Han, Z. AI BIM coordinator for non-expert interaction in building design using LLM-driven multi-agent systems. *Autom. Constr.* **2025**, *180*, 106563. <https://doi.org/10.1016/j.autcon.2025.106563>.
36. Zhang, G.; Lu, C.; Luo, Q. Application of Large Language Models in the AECO Industry: Core Technologies, Application Scenarios, and Research Challenges. *Buildings* **2025**, *15*, 1944. <https://doi.org/10.3390/buildings15111944>.
37. Babiuch, M.; Smutný, P. Benchmarking Large Language Models for Embedded Systems Programming in Microcontroller-Driven IoT Applications. *Future Internet* **2024**, *18*, 94. <https://doi.org/10.3390/fi18020094>.
38. Takyi-Annan, G.E.; Zhang, H. A Multivariate Analysis of the Variables Impacting the Level of BIM Expertise of Professionals in the Architecture, Engineering and Construction (AEC) Industries of the Developing World Using Nonparametric Tests. *Buildings* **2023**, *13*, 1606. <https://doi.org/10.3390/buildings13071606>.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.