fire

MDPI

Article

Algorithmic Evaluation of Fire Evacuation Efficiency Under Dynamic Crowd and Smoke Conditions

Hyunseok Kim [1], Sunnie Haam [2], Mintaek Yoo [1,*] and Woo Seung Song [3,*]

[1] Department of Civil and Environmental Engineering, Gachon University, 1342 Seongnam-daero, Sujeong-gu, Seongnam-si 13120, Republic of Korea; khsss1026@gachon.ac.kr
[2] Department of Fire Safety Engineering and Disaster Management, University of Seoul, 163, Seoulsiripdae-ro, Dongdaemun-gu, Seoul 02504, Republic of Korea; dandyhaam@uos.ac.kr
[3] Urban Life Network, Yongun-ro 1–3, 2F, Dae-dong, Dong-gu, Daejeon 34648, Republic of Korea
* Correspondence: mintaekyoo@gachon.ac.kr (M.Y.); songws@urbanlife.or.kr (W.S.S.)

Abstract

This study developed a fire evacuation simulation model for a six-level underground station to evaluate evacuation efficiency under both dynamic and static conditions, including structural damage, smoke propagation, and real-time crowd congestion. Two representative pathfinding algorithms, Dijkstra's and A*, were applied to analyze evacuation performance across eight fire scenarios occurring at different locations within the station. When only static factors were considered, both algorithms yielded identical maximum evacuation times, indicating comparable performance. However, the A* algorithm exhibited a significantly shorter computation time than Dijkstra's, demonstrating higher operational efficiency. When dynamic variables such as real-time congestion and smoke-induced visibility reduction were introduced, the maximum evacuation times varied irregularly between the two algorithms. This outcome suggests that, under dynamic fire conditions, route guidance based solely on current information rather than predictive modeling may lead to suboptimal evacuation outcomes. Therefore, this study emphasizes the importance of establishing a predictive disaster management system capable of forecasting fire and smoke propagation, as well as a centralized control system that can dynamically distribute evacuees to enhance evacuation efficiency in deep underground stations.

Keywords: fire evacuation; dynamic crowd; smoke propagation; Dijkstra algorithm; A* algorithm; evacuation efficiency; predictive control system

Check for updates

Academic Editors: M. Amin Hariri-Ardebili and Sissy Nikolaou

Received: 2 November 2025
Revised: 5 January 2026
Accepted: 6 January 2026
Published: 9 January 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.

1. Introduction

Subway systems serve as essential infrastructure that comprehensively promotes the economic, environmental, and social development of cities by enhancing workforce mobility and urban productivity. Major metropolitan subway networks worldwide accommodate over four million passengers daily, functioning as critical public transportation facilities that sustain urban life [1]. According to the International Association of Public [2], subway systems have become a key component of urban transportation infrastructure, with their global expansion steadily increasing over the past two decades.

However, disasters occurring in underground stations pose severe challenges owing to their structural characteristics, which make evacuation more difficult compared with aboveground facilities [3]. During emergencies, such as fires or structural failures, rescue operations become particularly complicated, and the likelihood of casualties increases. These challenges are further exacerbated during peak commuting hours when passenger

congestion hinders the identification of safe evacuation routes. Owing to their enclosed and densely populated environments, subways are inherently vulnerable to various disasters—including fires, terrorism, earthquakes, flooding, and crowd crush incidents—that have led to significant casualties each year [4].

For instance, the 2003 Daegu subway fire in Korea caused 192 fatalities and 148 injuries, while a 2015 fire in the Washington Metro in the United States resulted in one death and two injuries due to smoke inhalation. Flood-related incidents have also been reported, such as the 2014 inundation of Lines 1, 2, and 4 in Busan, Korea, and the 2016 flooding of multiple stations on Line 4 in Wuhan, China. In terms of seismic events, the 1995 Kobe earthquake (Mw 7.3) led to the collapse of Daikai Station and damage to five other subway stations in Japan, while the 2015 Delhi earthquake (Mw 7.8) caused train stoppages and passenger entrapment. Moreover, between 1980 and 2022, more than 440 crowd crush incidents were reported globally, resulting in over 13,700 fatalities [5]. In addition, electric vehicle (EV) fires pose critical challenges in confined spaces such as underground EV charging facilities, where thermal runaway of high-energy lithium-ion batteries can lead to rapid smoke propagation and severe deterioration of visibility. Brzezinska and Bryant (2022) [6] investigated the impact of EV fires on evacuation safety through performance-based fire analysis, while Bolina et al. (2025) [7] demonstrated the potential for fire spread associated with EV-related fire scenarios. These previous studies underscore the necessity of investigating fire scenarios in underground structures, thereby reinforcing the relevance of the findings presented in this study.

Given these recurring disasters, identifying safe and efficient evacuation routes is crucial to minimizing human casualties. Consequently, research on disaster response and evacuation modeling has expanded significantly, incorporating crowd congestion and behavioral dynamics into simulation frameworks. Senanayake et al. (2024) [8] systematically reviewed pedestrian evacuation simulations, classifying major theories, tools, and applications of agent-based models (ABM). Lopez-Carmona and Paricio Garcia (2021) [9] proposed an adaptive guidance system based on mobile devices and environmental sensors, dynamically adjusting evacuation routes in real time to mitigate local overcrowding. Liu et al. [10] developed a 3D real-time crowd evacuation framework replicating the Sensory–Decision–Motor (SDM) flow using a 3D adaptive Social Force Model (SFM) combined with a personalized gait controller to reflect individual walking behavior and collision avoidance. Sun et al. [11] analyzed collective evacuation behavior, emphasizing the impact of leadership, decision-making, and cooperative movement on evacuation efficiency. Tong and Bode [12] simulated three evacuation strategies for assisting vulnerable pedestrians using the Social Force Model. Fukuda [13] developed an agent-based "Evacuate Now Button" simulation for flood evacuation, while Oliveira [14] conducted a large-scale NetLogo-based simulation involving 18,000 individuals to evaluate the stability and efficiency of crowd evacuation strategies. Similarly, Choi et al. [15] introduced a Unity-based real-time 3D crowd density simulation system to predict and prevent crush incidents. These studies demonstrate that considering crowd congestion is indispensable when developing evacuation simulations or disaster mitigation strategies.

To address the limitations of traditional static simulations, several studies have incorporated dynamic and cognitive factors into evacuation models. Xiong et al. [16] proposed the Dynamic Indoor Field Model (DIFM), integrating building geometry, sensor data, and human behavioral responses. Dubey et al. [17] introduced a Cognitive Agent-Based Model that simulates human perception and decision-making under disaster conditions by fusing multisource information, improving evacuation route planning and emergency guidance systems. Pyakurel [18] summarized optimization techniques for evacuation planning using static and dynamic network flow models, comparing mathematical algo-

rithms and heuristic approaches. In addition, studies employing simulation tools such as PyroSim and Pathfinder have been actively conducted to evaluate fire and smoke analysis results in underground spaces by coupling them with evacuation simulations performed using Pathfinder [19,20].

Pathfinding algorithms play a crucial role in evacuation simulation models by determining optimal evacuation routes under complex and evolving conditions. Numerous studies have been conducted to develop and refine such algorithms. Pillac et al. [21] proposed a Conflict-Based Path-Generation (CPG) heuristic that integrates dynamic routing and scheduling under capacity constraints, successfully applied to a large-scale flood evacuation scenario involving 70,000 individuals. Bi et al. [22] comprehensively reviewed evacuation algorithms and systems for confined spaces, classifying methodologies and proposing AI-based evacuation simulations using big data. Ronchi et al. [23] reviewed behavior-based algorithms that simulate human decision-making during disasters. Lim et al. [24] developed a time-expanded multi-commodity network flow (MCNF) optimization model for hurricane evacuations, combining the Dijkstra algorithm for shortest-path estimation with the Ford–Fulkerson algorithm for maximum flow computation. An et al. [25] compared three representative algorithms—A*, SFM, and PPO—in single-level structures, highlighting discrepancies between theoretically optimal routes and behaviorally realistic paths. Halim et al. [26] proposed the K-Means Indoor Evacuation Assessment Algorithm for high-rise buildings. Additional studies have also proposed various algorithmic approaches for evacuation planning [27–30].

While numerous studies have incorporated crowd congestion and dynamic disaster characteristics into evacuation simulations, few have specifically addressed underground stations that feature complex vertical structures and constrained pathways.

In addition, the Dijkstra algorithm and the A* algorithm employed in this study have been extensively investigated as optimal evacuation route-finding algorithms. However, despite being representative path-finding algorithms, there is a lack of studies that provide a clear evaluation of their evacuation performance when static and dynamic variables are applied separately. Accordingly, this study does not aim to achieve a high-fidelity reproduction of fire dynamics themselves; rather, it focuses on analyzing the behavior of route-finding algorithms and evaluating their relative performance under static and dynamic conditions. Therefore, this study develops an evacuation simulation model for a six-level underground station that distinguishes between static and dynamic variables, applying two representative pathfinding algorithms, Dijkstra's and A*, to comparatively evaluate their performance. The findings aim to establish a methodological foundation for selecting appropriate evacuation algorithms in future predictive, dynamically adaptive evacuation systems.

2. Research Methodology

2.1. Algorithm

To evaluate algorithmic performance under static and dynamic conditions, this study applied two representative pathfinding algorithms, Dijkstra's algorithm and the A* algorithm, which are widely recognized as optimal route search methods. Pathfinding algorithms are generally classified into two categories: uninformed (blind search) and informed (heuristic-based) search methods.

The Dijkstra algorithm belongs to the uninformed search category. It systematically explores all possible paths from the start node to the destination node and identifies the shortest path by comparing the cumulative distances of each route. Because it evaluates every possible case, the resulting shortest path is guaranteed to be the true optimal

route [31]. However, this exhaustive search approach requires significant computational time, especially in large or complex networks.

In contrast, the A* algorithm is a heuristic-based search method that combines path cost and heuristic estimation to accelerate route discovery. Over the past decade, the A* algorithm has been continuously refined for improved stability and computational efficiency. While it can find near-optimal routes faster than uninformed methods, it does not always guarantee the absolute shortest path.

The primary difference between Dijkstra's and A* lies in the use of a heuristic function. Common heuristic functions include Manhattan distance, diagonal distance, and Euclidean distance, among others. Algorithms that employ heuristics are categorized as informed searches, as they guide the search process toward the destination more efficiently. Whereas uninformed searches, such as Dijkstra's, explore all possible nodes, informed searches can significantly reduce computation time; however, their performance may vary depending on the problem structure and heuristic accuracy [32]. The Dijkstra and A* algorithms were proposed several decades ago and have since become canonical methods in the field of pathfinding [33,34]. In particular, the two algorithms exhibit a clear structural distinction based on whether heuristic functions are employed. This study aims to quantitatively compare and analyze how this structural difference influences performance within an underground station evacuation guidance model. Based on these findings, this study proposes future research directions for the development of new evacuation route-finding algorithms for underground stations.

In this study, both the Dijkstra and A* algorithms were implemented to ensure comparability between exhaustive and heuristic-based searches. To enhance computational efficiency in the three-dimensional underground station network, the Euclidean distance was adopted as the heuristic function for the A* algorithm. Under static conditions with time-invariant edge weights, the Euclidean-distance heuristic adopted in the A* algorithm is admissible and consistent, resulting in optimal evacuation paths identical to those obtained using Dijkstra's algorithm. The computational flowcharts of Dijkstra's and A algorithms are illustrated in Figure 1. The dashed boxes represent the heuristic function used in the A algorithm, highlighting the key difference from Dijkstra's algorithm, and the corresponding computational procedures are depicted in Figure 1.

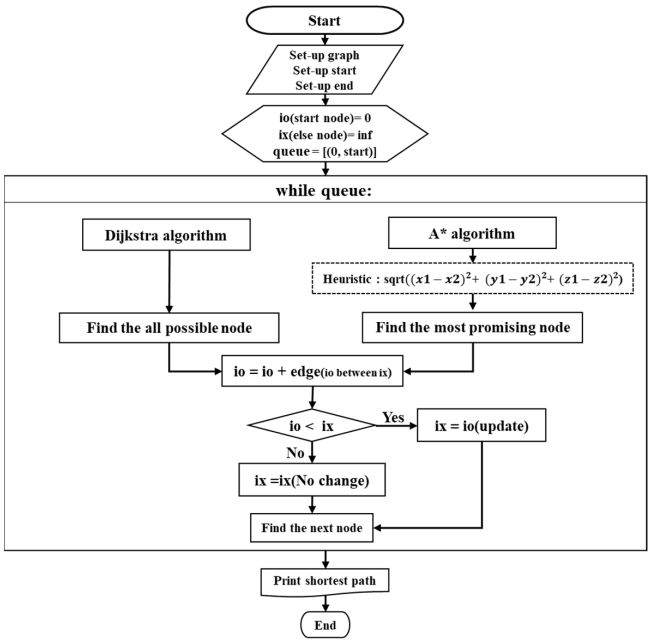

Figure 1. Flow Diagrams of the Dijkstra and A Pathfinding Algorithms.

2.2. Evacuation Simulation Model

An evacuation simulation model was developed for a six-level underground station to evaluate the performance of pathfinding algorithms under dynamic conditions, incorporating real-time crowd congestion and smoke propagation caused by fire. Python (3.12.0) was used as the programming language to implement the optimal evacuation route model, and the simulation procedure was designed as follows.

Before the simulation begins, several key parameters are predefined: current_count, transit_state, travel_time, and disaster condition. The variable current_count stores the initial number of passengers located at each node. The transit_state contains information about the number of passengers in transit, their movement time, starting node, and destination node. The travel_time parameter represents the edge value between two connected nodes, corresponding to the actual travel time in seconds. The disaster condition defines the weight adjustment caused by toxic gas and smoke generation from a fire event.

The computational procedure of the evacuation simulation model is as follows. Prior to executing the code, the number and locations of evacuees (current_count), movement information (transit_state), travel time (travel_time), and disaster conditions are initialized. Once the simulation begins, optimal evacuation routes are identified using route-finding algorithms, namely Dijkstra and A*. Evacuees stored in current_count are then transferred to transit_state, together with the corresponding travel time values representing the next movement segment. The transit_state records the origin node, destination node, number of evacuees, and remaining travel time.

At each iteration step (in seconds), the travel time stored in transit_state is reduced by one. When the travel time becomes zero or less, the evacuees in transit are considered to have arrived at the destination node and are moved back to current_count. If at least one evacuee remains in current_count, the route-finding algorithm is executed again to guide evacuees along updated optimal evacuation routes by accounting for real-time congestion levels and dynamic disaster conditions based on the information stored in transit_state. During this process, congestion occurrence is evaluated using the information of evacuees currently in transit, and when congestion is detected, a congestion weight is applied to the corresponding edge.

The operational sequence of the model that incorporates disaster propagation is as follows. Once the simulation begins, the program first identifies the disaster node and calculates the radius of smoke propagation to determine the nodes affected by the fire. To prevent multiple updates of the same edge, the model checks for duplication before modifying edge weights. Subsequently, the affected edges are updated by increasing their weights according to the calculated smoke concentration.

When the evacuation simulation starts, the passengers located at each current_node use either Dijkstra's or A* algorithm to determine the optimal evacuation path and identify their next node (next_node). The information regarding next_node, travel_time, and people_count is then recorded in transit_state. Based on the number of passengers currently in transit, the model updates the edge values every second by applying predefined crowd congestion factors. Simultaneously, the disaster state is updated according to the calculated smoke propagation speed and time intervals.

If both disaster-induced weights (from smoke and heat) and crowd congestion weights are applied to the same edge, the combined weight is computed as the product of both factors. For each iteration of the simulation, the travel_time value within transit_state decreases by one second. When travel_time becomes less than zero, the corresponding passengers are considered to have arrived at the next_node. The transit_state data are then transferred to current_count, which stores passengers who are not currently in motion.

As soon as a passenger's data are updated in current_count, the optimal next node is recalculated using the pathfinding algorithm.

The simulation continues iteratively until all passengers have successfully evacuated. Once complete evacuation is achieved, the model outputs the maximum evacuation time and optimal evacuation routes. The computational flow of this process is illustrated in Figure 2. The dashed boxes indicate conditional steps that can be activated or deactivated, as shown in Figure 2. The static disaster condition was implemented as a deterministic edge-weight amplification scheme, where predefined edges adjacent to disaster nodes were multiplied by a constant factor. The dynamic disaster condition was modeled as a time-dependent edge-weight update process based on a fixed propagation velocity, without probabilistic or empirical fire modeling.

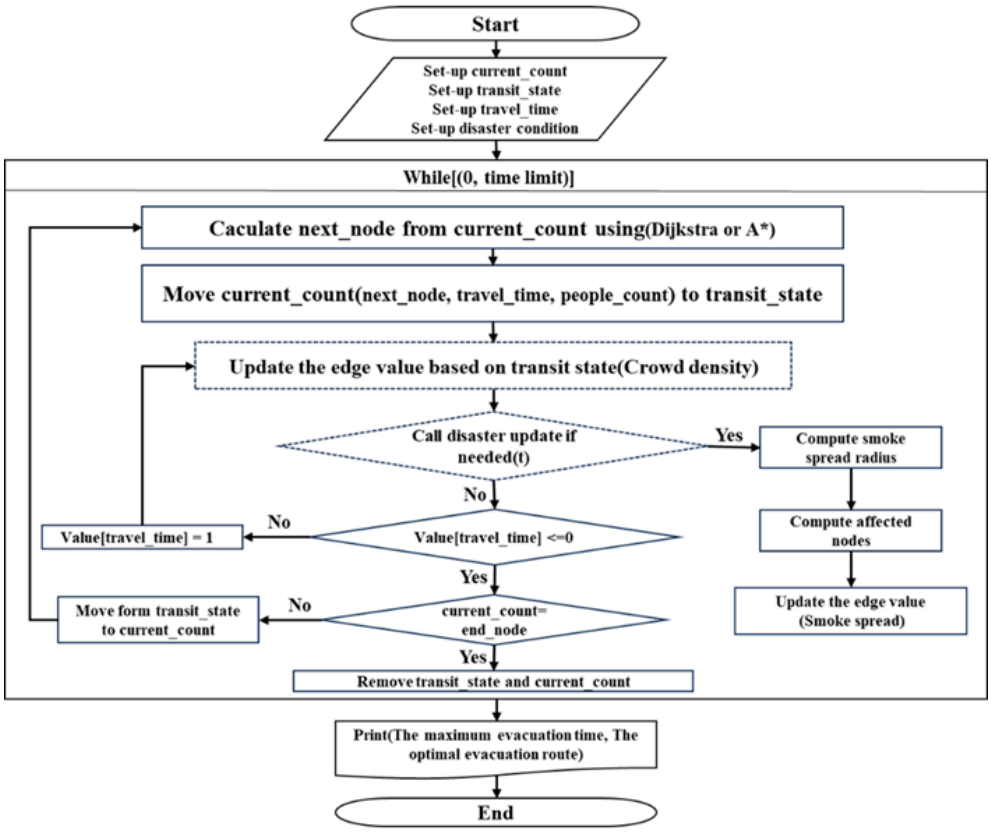

Figure 2. Flowchart of the Evacuation Simulation Model.

3. Analysis Condition

3.1. Underground Station Model

To conduct the evacuation simulation considering disaster variables and real-time crowd congestion, a six-level underground station located in Seoul was modeled. Based on actual station information, the model was constructed using horizontal nodes, vertical nodes, exit nodes, and edges, consisting of a total of 720 nodes and 2222 edges. Among them, there were 662 horizontal nodes, 56 vertical nodes, and 2 exit nodes, as summarized in Table 1.

For horizontal movement, the distance between adjacent nodes was set to 5 m. The platform included a total of 41 horizontal nodes, with an overall length of 200 m. For vertical connections, the ceiling height was assumed to be 5 m, and the stair inclination angle was set to 45 degrees, resulting in a total vertical length of 7 m. The distance between vertical nodes was defined as 3.5 m.

Table 1. Configuration of Nodes and Edges.

Configuration of Nodes and Edges			
Horizontal Nodes (Blue)	**Vertical Nodes (Red)**	**Exit Nodes (Green)**	**Edges (Gray)**
662	56	2	2222

In the station model, blue nodes represent horizontal sections, red nodes indicate stair (vertical) sections, and green nodes correspond to exits. Gray lines represent edges connecting the nodes. The overall geometric configuration of the modeled underground station is illustrated in Figure 3.

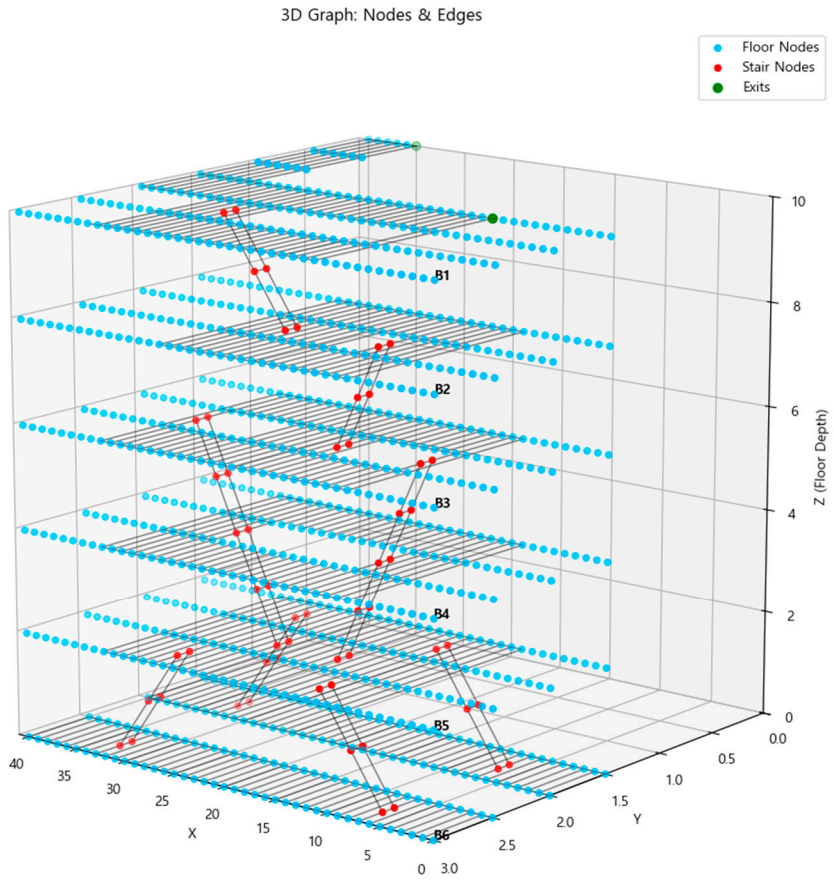

Figure 3. 3D Model of the Underground Station.

3.2. Pedestrian Evacuation Speed

To simulate passenger evacuation under assumed disaster conditions in the underground station, the horizontal and vertical movement speeds were determined based on the design guidelines of the Korean Ministry of Land, Infrastructure and Transport (MOLIT) [35]. The horizontal movement speed was set to 60 m/min, and the vertical movement speed was set to 15 m/min, as summarized in Table 2.

Table 2. Evacuation Speeds by Evacuation Factor.

Evacuation Factor	Evacuation Speed
Horizontal transportation (platform, concourse, corridor)	60 m/min
Vertical transportation (stairs, stationary, escalator)	15 m/min

To calculate the total evacuation time, the edge values were converted from distance (m) to time (s). For horizontal movement between nodes with a spacing of 5 m, the travel time was defined as 5 s. For vertical movement between nodes spaced 3.5 m apart, the travel time was set to 3.5 s. These converted edge values were used for the evacuation analysis.

3.3. Crowd Density Configuration

To incorporate real-time crowd congestion into the evacuation simulation, the minimum number of passengers that trigger congestion and the corresponding speed reduction rates were defined. According to the Korean Ministry of Land, Infrastructure and Transport (MOLIT) [35], the minimum evacuation capacity per unit width (m) is 80 persons/m·min for horizontal movement and 60 persons/m·min for vertical movement, as summarized in Table 3.

Table 3. Minimum Evacuation Capacity per Unit Width (m).

Evacuation Factor	Evacuation Capacity
Horizontal transportation (platform, concours, corridor)	80 persons/m·min
Vertical transportation (stairs, escalator)	60 persons/m·min

In this study, each horizontal node was modeled with an area of 5 m × 5 m, and each vertical node was modeled with an area of 3 m × 3.5 m. Using these dimensions, the minimum evacuation capacity without congestion was calculated based on the MOLIT standards. For horizontal transportation, 80 persons/m·min corresponds to 1.33 persons/m·s. Considering the horizontal node area of 25 m^2, the minimum evacuation capacity becomes approximately 33 persons per second per node. Therefore, when the number of passengers at a horizontal node exceeds 33, a reduction in walking speed occurs.

Similarly, for vertical transportation, 60 persons/m·min corresponds to 1 person/m·s. Considering the vertical node area of 10.5 m^2, the minimum evacuation capacity becomes approximately 11 persons per second per node. Thus, when more than 11 passengers occupy a vertical node, the walking speed begins to decrease.

Based on these minimum capacity thresholds, a speed reduction table was constructed using the relationship between occupant density (person/m^2) and walking speed presented by Zhou et al. [36], as shown in Figure 4. According to this relationship, when the occupant density is 1 person/m^2, the walking speed is approximately 1 m/s. As the density doubles to 2 persons/m^2, the walking speed decreases to 0.5 m/s, and when it triples to 3 persons/m^2, it decreases to approximately 0.4 m/s. Beyond this level, the walking speed falls below 0.3 m/s.

In this study, these reduction ratios were applied to define the crowd congestion conditions. For horizontal nodes, no reduction in walking speed occurs when the number of passengers is 33 or fewer. When it exceeds 33, the walking speed is reduced by a factor of 2; exceeding 66 reduces it by a factor of 2.5; and exceeding 99 reduces it by a factor of 3. Likewise, for vertical nodes, no reduction occurs when the number of passengers is 11 or fewer. When the number exceeds 11, 22, and 33, the walking speed reduction factors are 2, 2.5, and 3, respectively. The speed reduction coefficients according to crowd congestion are summarized in Table 4.

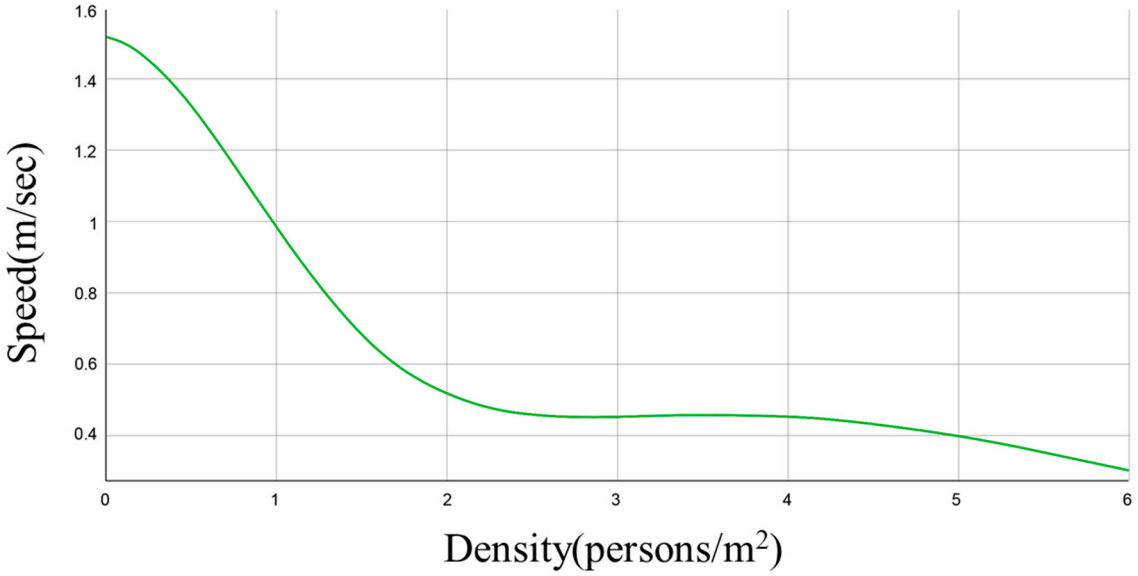

Figure 4. The fitting curve of the evacuation speed under different density in normal state.

Table 4. Walking Speed Reduction Table Based on Crowd Density.

Evacuation Factor	Population (Person)	Multiplier
Horizonal transportation (platform, concours, corridor)	33	2
Horizonal transportation (platform, concours, corridor)	66	2.5
Horizonal transportation (platform, concours, corridor)	99	3
Vertical transportation (stairs, escalator)	11	2
Vertical transportation (stairs, escalator)	22	2.5
Vertical transportation (stairs, escalator)	33	3

3.4. Disaster Parameter Setting

To evaluate the applicability of each algorithm under static and dynamic disaster conditions, the disaster variables were defined as follows. In this study, the static disaster scenario assumed structural damage. The edges adjacent to the disaster nodes were assigned a weight increased by a factor of 10, and the propagation of the disaster was restricted to prevent further spreading.

For the dynamic disaster scenario, a fire event was assumed, in which smoke propagation from the disaster node increased the weights of the adjacent edges over time. The smoke propagation speed was determined based on the tunnel fire experiment conducted by Hu et al. [37], in which the average smoke velocity was reported to be approximately 1 m/s under general conditions. Accordingly, a smoke propagation speed of 1 m/s was applied in this study.

Although smoke propagation can vary depending on fuel type, ventilation conditions, structural geometry, and heat release rate, this study applied a simplified model to focus on verifying the algorithmic adaptability under static and dynamic conditions. In the dynamic disaster scenario, the disaster propagated outward from the fire-origin node at a velocity of 1 m/s, with the corresponding edge weights doubling as time progressed. Once an

edge was updated, it was not modified again to avoid duplicate updates. The propagation proceeded from the initial disaster node to adjacent nodes.

The number of disaster nodes was set to two for all cases, and the parameters of the disaster variables are summarized in Table 5.

Table 5. Disaster Parameter Condition Table.

Disaster Types	Number of Disaster Nodes	Propagation Speed	Multiplier
Static Disaster (Structural Damage)	2	-	10
Dynamic Disaster (Smoke Propagation)	2	1 m/s	2

3.5. Evacuee Distribution

To determine the total number of evacuees for the simulation, the population during the peak commuting hour was applied, representing the most congested condition. The simulation assumed that a train arrived at the upbound platform, where passengers were waiting for boarding. The upbound platform consisted of 82 nodes, numbered from 0 to 81, and the downbound platform also comprised 82 nodes, numbered from 82 to 163.

According to the Korea Railroad Corporation (KORAIL, 2025), the total number of boarding passengers during the peak hour across 34 stations on a single Seoul subway line was 1,080,603. To account for conservative conditions, a 10% increase was applied, setting the total number of waiting passengers at 1200. These passengers were evenly divided, with 600 assigned to the upbound platform and 600 to the downbound platform.

Additionally, for the upbound platform where the arriving train was assumed, the number of passengers inside the train was also included. Based on a full capacity of 100%, approximately 160 passengers occupy each train car. In this study, a conservative congestion rate of 110% was applied, corresponding to 175 passengers per car. Given that the train operated with eight cars, approximately 1400 additional passengers were allocated to the upbound platform.

In total, 2000 passengers were distributed on the upbound platform and 600 passengers on the downbound platform. For the upbound platform, 25 passengers were assigned to each node from Node 0 to Node 25, and 24 passengers were assigned to each node from Node 26 to Node 81. For the downbound platform, 8 passengers were assigned to each node from Node 82 to Node 107, and 7 passengers were assigned to each node from Node 108 to Node 163.

The overall distribution of passengers between the upbound and downbound platforms is summarized in Table 6, and the detailed passenger allocation for each node is presented in Table 7.

Table 6. Evacuee Distribution on Upbound and Downbound Platforms.

Upbound Platform	Downbound Platform	Total (Person)
2000	600	2600

Table 7. Evacuee Distribution by Node.

Upbound Platform		Downbound Platform	
Node 0~25	**Node 26~81**	**Node 82~107**	**Node 108~163**
25	24	8	7

3.6. Analysis Case

In this study, four variable update models and eight disaster-location cases were applied to analyze the evacuation performance of each algorithm under static and dynamic conditions. A six-level underground station located in Seoul was modeled, and both the Dijkstra and A* algorithms were applied to determine the maximum evacuation time and assess their algorithmic suitability.

The variable update models consisted of three types: structural damage, fire smoke, and real-time crowd congestion. Among these, structural damage was classified as a static variable, whereas fire smoke and real-time congestion were classified as dynamic variables. By combining these variables, a total of four analysis models were established. Model 1 considered only structural damage, while Model 2 considered only fire smoke. Model 3 and Model 4 were combinations of two variables: Model 3 combined structural damage and real-time crowd congestion, and Model 4 combined fire smoke and real-time crowd congestion.

Additionally, to compare the maximum evacuation times according to the disaster locations, eight cases were analyzed. In Case 1, the disaster occurred at the center of the platform where the train arrived. In Case 2, it occurred at the center of the opposite platform. Case 3 assumed a disaster at the edge of the platform where the train arrived, and Case 4 assumed one at the edge of the opposite platform. In Cases 5 through 8, disasters occurred on stairways, with four stair locations assigned as individual disaster nodes.

The variable combinations for each analysis model are summarized in Table 8. Applied conditions are indicated by 'O', while non-applied conditions are denoted by '–'. And the disaster-location cases are summarized in Table 9.

Table 8. Simulation Models by Variable Type.

Model	Static Disaster (Structural Damage)	Dynamic Disaster (Smoke Propagation)	Real-Time Crowd Density
Model 1	O	-	-
Model 2	-	O	-
Model 3	O	-	O
Model 4	-	O	O

Table 9. Simulation Cases by Disaster Location.

Case	Disaster Location	Disaster Nodes
1	Center of Train Platform	20, 61
2	Center of Opposite Platform	92, 133
3	Side of Train Platform	8, 41
4	Side of Opposite Platform	82, 123
5	Stairs at Train Platform 1	1008, 1009
6	Stairs at Train Platform 2	1010, 1011
7	Stairs at Opposite Platform 1	1000, 1001
8	Stairs at Opposite Platform 2	1002, 1003

4. Results

The evacuation performance of each algorithm was analyzed by comparing the maximum evacuation times under static and dynamic variable conditions. In addition, evacuation simulations were conducted for different disaster locations using Dijkstra's algorithm

and the A* algorithm. A total of 64 simulations were performed, combining four static and dynamic models, eight disaster-location cases, and two algorithms, and the results were analyzed accordingly.

In Model 1, which considered only the static variable (structural damage), the maximum evacuation times of the two algorithms were identical regardless of the disaster location. This indicates that when dynamic variables are not applied, both algorithms exhibit the same performance even in a large-scale model consisting of approximately 760 nodes. However, because Dijkstra's algorithm is an uninformed (blind search) method, whereas the A* algorithm is an informed search method that uses a heuristic function, the computation time of Dijkstra's algorithm was approximately 12.7 times longer than that of the A* algorithm. Although both algorithms produced identical maximum evacuation times in Model 1, the A* algorithm demonstrated significantly higher computational efficiency. This result verifies that the use of heuristic functions leads to a distinct difference in computational time during the pathfinding process. The evacuation simulation using the proposed code was performed on a workstation equipped with an AMD Ryzen 9 9950X 16-core processor, an NVIDIA GeForce RTX 3050 GPU, and 16 GB RAM. The comparison of computation times between the two algorithms in Model 1 is summarized in Table 10.

Table 10. Calculation Time of Model1.

Case	Model 1	
	Maximum Evacuation Time	
	Dijkstra Algorithm (s)	A* Algorithm (s)
1	296.924859	25.196902
2	309.113998	26.036973
3	264.656259	23.171157
4	254.151283	21.754981
5	402.593697	30.097244
6	398.779278	29.213384
7	384.739460	28.312938
8	392.070627	29.211842
Average	337.88	26.62

In Model 2, which applied the dynamic variable of fire-induced smoke, the maximum evacuation times varied between the two algorithms depending on the disaster location. In Case 1, both algorithms produced identical maximum evacuation times, but in other cases, the results differed. Specifically, Dijkstra's algorithm showed shorter maximum evacuation times in Cases 2, 6, and 8, whereas the A* algorithm produced shorter times in Cases 3, 4, 5, and 7.

In Model 3, which combined the static variable (structural damage) and the dynamic variable (real-time crowd congestion), Dijkstra's algorithm resulted in shorter maximum evacuation times in Cases 4, 6, and 7, while the A* algorithm showed shorter times in Cases 1, 2, 3, 5, and 8.

Finally, in Model 4, which combined the dynamic variables of fire smoke and real-time crowd congestion, the two algorithms produced identical maximum evacuation times in Case 7. However, Dijkstra's algorithm showed shorter evacuation times in Cases 1, 3, 4, 6, and 8, while the A* algorithm performed better in Cases 2 and 5. The summary of maximum evacuation time results for each case is presented in Table 11.

Table 11. The result of each cases about Maximum evacuation time.

Model 1		
Case	**Maximum Evacuation Time**	
	Dijkstra Algorithm (s)	**A* Algorithm (s)**
1	409	409
2	443	443
3	399	399
4	399	399
5	498	498
6	484	484
7	497	497
8	483	483
Model 2		
Case	**Maximum Evacuation time**	
	Dijkstra Algorithm (s)	**A* Algorithm (s)**
1	740	740
2	788	792
3	825	774
4	744	734
5	880	736
6	747	750
7	882	738
8	745	752
Model 3		
Case	**Maximum Evacuation time**	
	Dijkstra Algorithm (s)	**A* Algorithm (s)**
1	1103	997
2	1015	1006
3	1056	1026
4	987	996
5	1303	1276
6	1020	1052
7	1003	1007
8	1034	1016
Model 4		
Case	**Maximum Evacuation time**	
	Dijkstra Algorithm (s)	**A* Algorithm (s)**
1	1859	1898
2	1847	1802
3	1857	2028
4	1815	1914

Table 11. *Cont.*

5	1891	1864
6	1862	1958
7	1876	1876
8	1833	1817

5. Discussion

In this study, the evacuation performance of Dijkstra's algorithm and the A* algorithm was analyzed by comparing their maximum evacuation times under static and dynamic variables, as well as different disaster locations, using a six-level underground station model. Four variable combination models were established by combining one static variable and two dynamic variables, and eight disaster-location cases were applied to both algorithms, resulting in a total of 64 simulations.

In Model 1, which considered only the static variable (structural damage), the maximum evacuation times of both algorithms were identical regardless of the disaster location. This indicates that for a model consisting of approximately 760 nodes, both algorithms produced equivalent performance in terms of evacuation time. However, the computation times of the two algorithms differed significantly. The average computation time for Dijkstra's algorithm was 337.88 s, whereas that of the A* algorithm was 26.62 s. Dijkstra's algorithm, as an uninformed (blind search) method that explores all possible routes, guarantees the optimal solution but requires significantly longer computation time. Conversely, the A* algorithm, an informed search method utilizing a heuristic function, requires less computation time but does not always guarantee the absolute optimal solution. In this study, when only static variables were considered, both algorithms achieved identical evacuation performance, but the A* algorithm was found to be more efficient owing to its shorter computation time.

The equivalence of evacuation times between Dijkstra's and A* algorithms under static hazard conditions can be attributed to the time-invariant nature of edge weights. In this case, the Euclidean-distance heuristic used in the A* algorithm is admissible and consistent, ensuring that A* identifies the same optimal path as Dijkstra's algorithm.

In contrast, under dynamic hazard conditions, such as smoke propagation and real-time crowd congestion, edge costs evolve over time. Although both algorithms compute optimal routes based on current information, neither can anticipate future changes in hazard intensity or congestion. As a result, the optimality equivalence observed under static conditions is no longer preserved, leading to different evacuation outcomes depending on disaster location and timing.

In contrast, when dynamic variables were introduced, the results varied according to the disaster locations. In Model 2, which considered fire-induced smoke, the maximum evacuation times of both algorithms differed in all cases except Case 1. Dijkstra's algorithm produced shorter evacuation times in Cases 2, 6, and 8, whereas the A* algorithm yielded shorter times in Cases 3, 4, 5, and 7. Similar trends were observed in Model 3 (structural damage combined with real-time crowd congestion) and Model 4 (fire smoke combined with real-time crowd congestion). In Model 3, Dijkstra's algorithm showed shorter evacuation times in Cases 4, 6, and 7, while the A* algorithm performed better in Cases 1, 2, 3, 5, and 8. In Model 4, both algorithms exhibited identical evacuation times in Case 7, but Dijkstra's algorithm produced shorter times in Cases 1, 3, 4, 6, and 8, while the A* algorithm performed better in Cases 2 and 5. The maximum evacuation time for each algorithm case is illustrated in Figure 5, indicating that the performance of the algorithms exhibits random variations across different cases.

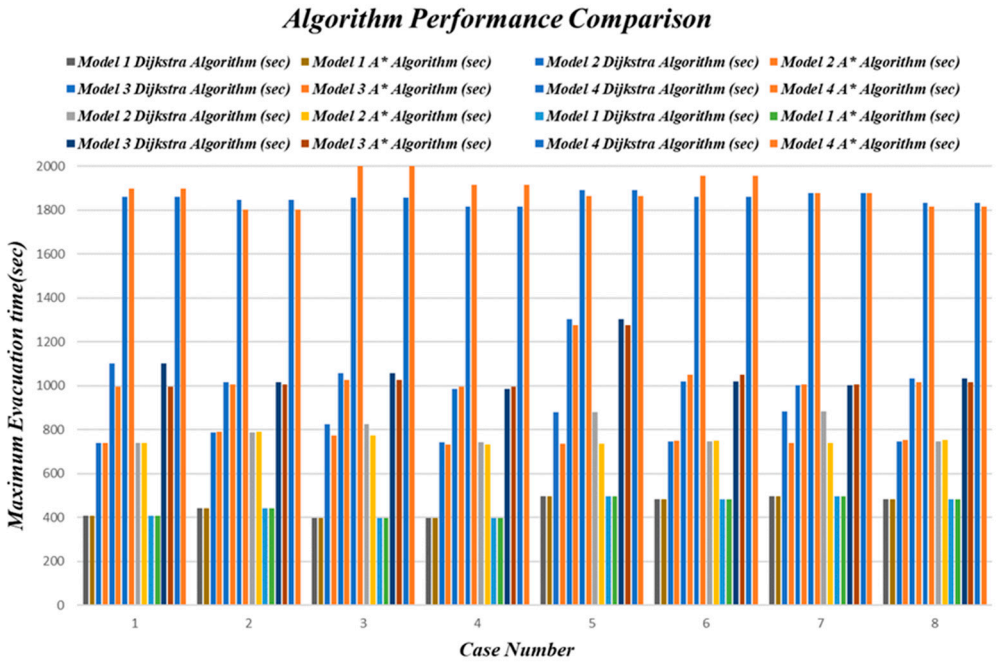

Figure 5. Algorithm Performance Comparison.

The reason for the differences between the algorithms under dynamic conditions lies in the uncertainty of future disaster evolution. Even though Dijkstra's algorithm guarantees the optimal route by exploring all possible paths, its maximum evacuation time was sometimes longer than that of the A* algorithm in dynamic disaster scenarios. This is because, under fire-induced smoke propagation, the situation may change over time; even if evacuees are guided through the optimal route at the current moment, the rapid spread of smoke may later render that route slower or less safe. Similarly, when real-time crowd congestion is applied, the results show a comparable pattern. Although the pathfinding algorithm guides evacuees through the optimal route based on current congestion data, it may inadvertently lead to higher congestion in subsequent time steps. This phenomenon resembles a traffic navigation system: when one highway becomes congested, rerouting all vehicles to an alternative "optimal" route can eventually cause that new route to become more congested and slower than the original.

Therefore, when considering dynamic variables such as fire smoke and real-time congestion, an algorithm alone is insufficient to ensure efficient evacuation. A centralized control system is required to predict congestion levels and future hazard developments.

The centralized control system refers to an integrated platform where data from sensors, digital signage, and environmental detectors are transmitted to a central controller that evaluates the global crowd state and provides optimal evacuation guidance [38]. However, simply calculating the current optimal route based on real-time data does not account for future uncertainties or potential secondary congestion. This implies that future evacuation management should integrate predictive disaster modeling—using AI-based simulations—to forecast hazard evolution and guide evacuees proactively.

When incorporating real-time crowd congestion into an optimal evacuation route model, evacuees should not all be directed through a single optimal route. Instead, a centralized control system should distribute passengers across multiple routes to prevent excessive congestion, reduce total evacuation time, and mitigate the risks of crush and secondary accidents. Although this study did not employ advanced fire simulation software such as FDS to reproduce detailed fire dynamics, it intentionally used simplified dynamic variables to evaluate algorithmic adaptability under varying conditions.

The centralized evacuation control system discussed in this study is defined as a conceptual system that integrates multiple sensor-based data sources to enable coordinated evacuation guidance. Key data inputs include information on evacuee numbers, locations, and real-time movement trajectories obtained through CCTV-based people counting systems or indoor positioning technologies, which allow continuous monitoring of crowd distribution and pedestrian flow within the station. Such information can be used to anticipate localized congestion and to intentionally distribute evacuees across multiple evacuation routes. In addition, fire-related disaster information collected from fire detectors and environmental sensors constitutes a critical input to the centralized control system. In this context, fire hazard propagation is represented by indicators such as smoke concentration, temperature, and smoke-induced visibility reduction. These variables can be used to quantitatively assess time-dependent hazard severity and to evaluate the safety of evacuation routes. By integrating these heterogeneous sensor data, the centralized evacuation control system enables predictive evacuation strategies that consider near-future congestion and hazard evolution, rather than relying solely on reactive, real-time route guidance. The present study does not aim to design or implement such a system, but rather to demonstrate its necessity based on the observed limitations of algorithm-based evacuation guidance under dynamic conditions.

Overall, the results of 64 simulations demonstrated that the performance of Dijkstra's and A* algorithms varied depending on disaster location when dynamic variables were considered. Based on these findings, it is concluded that future evacuation models should move beyond reactive approaches that only respond to current disaster states and instead adopt predictive models capable of estimating future hazard propagation. Furthermore, when real-time crowd congestion is incorporated, centralized control systems that dynamically distribute evacuees across multiple routes will be essential to minimize delays, prevent crush incidents, and enhance overall evacuation safety in underground stations.

6. Limitation

In this study, analyses were conducted to identify a suitable evacuation route-finding algorithm for the development of an optimal evacuation route model and to examine the limitations and underlying causes associated with each algorithm. Accordingly, the analyses were performed based on simplified assumptions of disaster propagation. However, the proposed simulation model has several limitations. The smoke propagation mechanism was simplified, and various fire dynamic effects were not considered. In addition, behavioral diversity of evacuees and environmental factors such as ventilation were not taken into account. Furthermore, not all detailed components of the underground station were explicitly modeled, and stair geometries were simplified in terms of inclination angle and spacing. A formal mesh-independence or discretization sensitivity analysis was not conducted in this study, as the focus was placed on comparative algorithmic behavior rather than on mesh-converged evacuation time predictions.

In future studies, these limitations will be addressed by developing evacuation algorithms capable of responding to dynamic disaster conditions. Moreover, validated fire simulation tools, such as the Fire Dynamics Simulator (FDS), will be integrated to develop an optimized centralized evacuation control algorithm. Accordingly, the conclusions of this study should be interpreted as algorithmic performance characteristics rather than physically accurate evacuation predictions under real fire conditions.

7. Conclusions

This study analyzed evacuation performance by conducting 64 simulations that combined three variables into four models, eight disaster-location cases, and two pathfinding

algorithms for a six-level underground station. When only the static variable of structural damage was considered, the maximum evacuation times obtained from Dijkstra's algorithm and the A* algorithm were identical across all disaster locations, indicating equivalent performance. However, the A* algorithm required less computation time than Dijkstra's algorithm, demonstrating higher efficiency.

When dynamic variables such as fire-induced smoke and real-time crowd congestion were applied, the maximum evacuation times varied between the two algorithms depending on the disaster location. This result implies that under dynamic conditions, algorithm performance does not have a consistent relationship with evacuation time. To effectively reduce evacuation time during disasters, two critical functions must be incorporated into evacuation modeling.

First, instead of calculating the optimal evacuation route solely from real-time disaster data, the model should predict disaster propagation and guide evacuees based on the forecasted spread. In principle, Dijkstra's algorithm, which guarantees the optimal path, should produce equal or shorter evacuation times than the A* algorithm, which does not always guarantee optimality. However, in several dynamic scenarios, the A* algorithm resulted in shorter maximum evacuation times. This indicates that the conventional approach of continuously recalculating and guiding passengers along the "optimal" real-time route can sometimes produce adverse effects, as the future spread of smoke or structural hazards is not reflected. Therefore, future evacuation systems must incorporate predictive modeling that anticipates disaster progression and provides pre-optimized route guidance rather than relying solely on real-time information.

Second, when real-time crowd congestion is considered, evacuees tend to be redirected away from congested routes toward new "optimal" paths. However, this rerouting can cause secondary congestion as subsequent evacuees follow the newly assigned route. The A* algorithm, as an informed search method, occasionally demonstrated shorter evacuation times than Dijkstra's algorithm, not because of higher accuracy, but due to unintended dispersion effects that distributed evacuees across multiple routes. Nevertheless, it would be inappropriate to rely on such incidental effects for evacuation planning. To achieve intentional and controlled dispersion, a centralized control system is required. Such a system, powered by artificial intelligence, should dynamically guide evacuees along distributed routes from the start of evacuation, reflecting predicted congestion levels in advance rather than directing all passengers along a single optimal path.

In disaster situations within underground stations, both real-time disaster information and real-time crowd congestion must be considered to ensure safe evacuation. However, evacuation models that rely solely on instantaneous data may lead to performance degradation due to the unpredictability of evolving hazards. Therefore, future research should focus on developing predictive, AI-driven models that incorporate dynamic variables and employ centralized control systems to manage crowd distribution effectively.

In this study, two algorithms were applied under static and dynamic variables and different disaster locations, resulting in 64 simulations that derived the maximum evacuation times. When only the static variable was applied, both algorithms exhibited identical performance in a network of approximately 760 nodes. Under dynamic conditions, however, their performance was inconsistent, indicating that improving evacuation efficiency requires not merely selecting a better algorithm but developing a new system architecture.

Future studies should establish a crowd-dispersal evacuation model using a centralized control system that incorporates real-time congestion data. Such a system could reduce crowding during peak hours and enhance the resilience of subway stations against disasters. In conclusion, this study compared algorithmic performance under static and dynamic disaster scenarios and proposed future directions for the development of predic-

tive, adaptive evacuation models. The findings highlight the importance of developing enhanced evacuation strategies that can contribute to safer and more resilient underground transportation systems.

Author Contributions: H.K.: Writing—original draft, methodology. S.H.: conceptualization, visualization. M.Y.: Writing—review and editing, conceptualization, methodology, W.S.S.: Writing—review and editing, supervision, project administration. The authors confirm that this work has not been published before. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Korea Agency for Infrastructure Technology Advancement (KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (Development of Fire Spread Prevention and Evacuation Response Technologies, RS-2025-02653713) and the 2025 Fire Agency Fund through the Korea Institute for Advancement of Technology (KIAT) as part of the "Development of Fire Response Technologies for Electric-Based Mobility Facilities and Components" project (Project No. RS-2024-00406709, Development of Early Fire Response Suppression System for Electric Vehicle Fires in Mechanical Parking Facilities and Related Fire Safety Standards). The authors gratefully acknowledge this financial support.

Data Availability Statement: Data are contained within the article.

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

Abbreviations

g(n)	Actual travel cost from the start node to node n
h(n)	Heuristic cost estimated from node n to the nearest exit
f(n)	Total estimated cost used in the A* algorithm (f = g + h)
A*	A-star pathfinding algorithm
FDS	Fire Dynamics Simulator
SFM	Social Force Model

References

1. Lin, D.; Broere, W.; Cui, J. Metro systems and urban development: Impacts and implications. *Tunn. Undergr. Space Technol.* **2022**, *125*, 104509. [CrossRef]
2. UITP (International Association of Public Transport). *World Metro Figures 2021*; UITP (International Association of Public Transport): Brussels, Belgium, 2022; p. 1.
3. Samadi, H.; Hassanpour, J.; Rostami, J. Prediction of earth pressure balance for EPB-TBM using machine learning algorithms. *Int. J. Geo-Eng.* **2023**, *14*, 21. [CrossRef]
4. Yu, H.; Wang, Y.; Qiu, P.; Chen, J. Analysis of natural and man-made accidents happened in subway stations and trains: Based on statistics of accident cases. *MATEC Web Conf.* **2019**, *272*, 01031. [CrossRef]
5. Weng, W.; Wang, J.; Shen, L.; Song, Y. Review of analyses on crowd-gathering risk and its evaluation methods. *J. Saf. Sci. Resil.* **2023**, *4*, 93–107. [CrossRef]
6. Brzezinska, D.; Bryant, P. Performance-based analysis in evaluation of safety in car parks under electric vehicle fire conditions. *Energies* **2022**, *15*, 649. [CrossRef]
7. Bolina, F.L.; Fachinelli, E.G.; Rodrigues, J.P.C. Analysis of building structures subjected to electric vehicle fires. *J. Build. Eng.* **2025**, *107*, 112769. [CrossRef]
8. Senanayake, G.P.; Kieu, M.; Zou, Y.; Dirks, K. Agent-based simulation for pedestrian evacuation: A systematic literature review. *Int. J. Disaster Risk Reduct.* **2024**, *111*, 104705. [CrossRef]
9. Lopez-Carmona, M.A.; Paricio Garcia, A. CellEVAC: An adaptive guidance system for crowd evacuation through behavioral optimization. *Saf. Sci.* **2021**, *141*, 105215. [CrossRef]
10. Liu, X.; Zhou, T.; Kang, H.; Ma, J.; Wang, Z.; Huang, J.; Weng, W.; Lai, Y.-K.; Li, K. RESCUE: Crowd evacuation simulation via controlling SDM-united characters. *arXiv* **2025**, arXiv:2507.20117. [CrossRef]
11. Sun, H.; Han, G.; Zhang, X.; Ruan, X. Grasping emergency dynamics: A review of group evacuation techniques and strategies in major emergencies. *J. Saf. Sci. Resil.* **2025**, *6*, 1–20. [CrossRef]

12. Tong, Y.; Bode, N.W.F. Simulation investigation on crowd evacuation strategies for helping vulnerable pedestrians at different stages of egress. *Int. J. Disaster Risk Reduct.* **2023**, *84*, 103479. [CrossRef]
13. Fukuda, T. Simulation-based analysis of evacuation information sharing systems using geographical data. *Int. J. Adv. Comput. Sci. Appl.* **2024**, *15*, 90–99. [CrossRef]
14. Oliveira, H.S. Optimizing crowd evacuation: Evaluation of strategies for safety and efficiency. *J. Reliab. Intell. Environ.* **2025**, *11*, 223–241. [CrossRef]
15. Choi, C.-H.; Park, S.-W.; Park, J.; Kim, J.-H.; Kim, J.-S.; Yang, H.-S.; Lee, B.-E.; Jung, S.-H.; Sim, C.-B. Prediction and prevention of crowd-crush accidents using crowd-density simulation based on Unity engine. *Discov. Appl. Sci.* **2025**, *7*, 33. [CrossRef]
16. Xiong, Q.; Zhu, Q.; Du, Z.; Zhu, X.; Zhang, Y.; Niu, L.; Zhou, Y. A dynamic indoor field model for emergency evacuation simulation. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 104. [CrossRef]
17. Dubey, R.K.; Sohn, S.S.; Hoelscher, C.; Kapadia, M. Cognitive agent-based simulation model for improving disaster response procedures. *arXiv* **2020**, arXiv:1910.00767.
18. Pyakurel, U. Network flow models for evacuation planning. *J. Inst. Eng.* **2018**, *13*, 90–101. [CrossRef]
19. Li, X.; Chen, R.; Zhu, Y.; Jim, C.Y. Emergency fire evacuation simulation of underground commercial street. *Simul. Model. Pract. Theory* **2024**, *134*, 102929. [CrossRef]
20. Zhang, N.; Liang, Y.; Zhou, C.; Niu, M.; Wan, F. Study on fire smoke distribution and safety evacuation of subway station based on BIM. *Appl. Sci.* **2022**, *12*, 12808. [CrossRef]
21. Pillac, V.; Van Hentenryck, P.; Even, C. A conflict-based path-generation heuristic for evacuation planning. *arXiv* **2013**, arXiv:1309.2693. [CrossRef]
22. Bi, H.; Gelenbe, E. A survey of algorithms and systems for evacuating people in confined spaces. *Electronics* **2019**, *8*, 711. [CrossRef]
23. Ronchi, R.Y.A.; Akbari, V.; Lawson, G. A systematic review of methodologies for human behavior modelling and routing optimization in large-scale evacuation planning. *Int. J. Disaster Risk Reduct.* **2024**, *110*, 104638. [CrossRef]
24. Lim, G.J.; Baharnemati, M.R.; Zangeneh, S.; Parsaei, H.R. A network flow based optimization approach for hurricane evacuation planning. In Proceedings of the International Conference on Value Chain Sustainability (ICOVACS) 2009, Louisville, KY, USA, 19–21 October 2009.
25. An, Z.; Wang, S.; Zhang, C.; Xing, A.; Yang, S. Comparative analysis of evacuation simulation algorithms for single-storey buildings. In Proceedings of the 2nd International Conference on Computing, Networks and Internet of Things (CNIOT 2021), Beijing, China, 20–22 May 2021; ACM: New York, NY, USA, 2021; pp. 1–4. [CrossRef]
26. Halim, A.H.A.; Samah, K.A.F.A. Validation of evacuation assessment algorithm in finding the best indoor evacuation model. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 340–347. [CrossRef]
27. Hasan, M.H.; Van Hentenryck, P. Large-scale zone-based evacuation planning—Part I: Models and algorithms. *Networks* **2021**, *77*, 127–145. [CrossRef]
28. Tokunaga, J.; Kikukawa, Y.; Ebara, H.; Ueda, N. Fast and accurate evacuation planning algorithm with Bayesian optimization. *ACM Trans. Intell. Syst. Technol.* **2025**, *16*, 23. [CrossRef]
29. Gerasimenko, E. Dynamic flow algorithm with intermediate storage for emergency evacuation allowing lane reversal based on incomplete intuitionistic fuzzy preference relation. *Open Transp. J.* **2024**, *18*, e18744478281682. [CrossRef]
30. Shahabi, K.; Wilson, J.P. Scalable evacuation routing in a dynamic environment. *Comput. Environ. Urban Syst.* **2018**, *67*, 29–40. [CrossRef]
31. Lawande, S.R.; Jasmine, G.; Anbarasi, J.; Izhar, L.I. A systematic review and analysis of intelligence-based pathfinding algorithms in the field of video games. *Appl. Sci.* **2022**, *12*, 5499. [CrossRef]
32. Foead, D.; Ghifari, A.; Kusuma, M.B.; Hanafiah, N.; Gunawan, E. A systematic literature review of A* pathfinding. *Procedia Comput. Sci.* **2021**, *179*, 507–514. [CrossRef]
33. Dijkstra, E.W. A note on two problems in connexion with graphs. In *Edsger Wybe Dijkstra: His Life, Work, and Legacy*; Association for Computing Machinery: Ney York, NY, USA, 2022; pp. 287–290. [CrossRef]
34. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [CrossRef]
35. Ministry of Land, Infrastructure and Transport (MOLIT). *Design Guidelines for Urban Railway Stations and Transfer Facilities (Notice No. 2022-78)*; MOLIT: Sejong, Republic of Korea, 2022.
36. Zhou, M.; Li, F.; Nie, G. Developing a revised social force model for earthquake emergency evacuation in underground structures. *Geomat. Nat. Hazards Risk* **2020**, *11*, 335–356. [CrossRef]

37. Hu, L.H.; Huo, R.; Yang, R.X.; He, W.H.; Wang, H.B.; Li, Y.Z. Full scale experiments on studying smoke spread in a road tunnel. *Fire Saf. Sci. Proc. Eighth Int. Symp.* **2005**, *8*, 1437–1447. [CrossRef]
38. Zhao, H.; Schwabe, A.; Schläfli, F.; Thrash, T.; Aguilar, L.; Dubey, R.K.; Schinazi, V.R. Fire evacuation supported by centralized and decentralized visual guidance systems. *Saf. Sci.* **2022**, *145*, 105451. [CrossRef]

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.