# Simulation Analysis of Evacuation Guidance Using Dynamic Distributed Signage

Akira Tsurushima[a]
*Intelligent Systems Laboratory, SECOM CO., LTD., Japan*

Keywords: Crowd Evacuation, Dynamic Evacuation Guidance, Distributed Problem Solving, Multi-Agent Simulation.

Abstract: Evacuation guidance systems must be adaptive and distributed in the unstable and harsh environment of disaster evacuation. Numerous dynamic evacuation guidance systems have been proposed and studied; however, few of them focus on the unstable evacuation environment that makes computer systems unreliable and malfunctioning. In this study, we introduce a distributed algorithm for a dynamic evacuation guidance system to ensure safe and efficient evacuation, even in the face of system component failures. The system is designed to be resilient, allowing it to continue functioning, providing effective evacuation guidance despite partial system malfunctions. Simulation experiments showed that the distributed system can provide more efficient evacuation guidance than static guidance systems. Furthermore, it correctly guided evacuees in situations where the target exit changed during the evacuation, showcasing the system's adaptability and effectiveness in handling unforeseen challenges, including system failures.

## 1 INTRODUCTION

Numerous studies have been conducted on crowd evacuations, and many devices and systems have been proposed and developed to support efficient evacuation. Several researchers are currently studying dynamic evacuation guidance systems that provide efficient advice on evacuation paths to evacuees considering the dynamic evacuation environments in real time. These systems typically include sensors and interface devices. Sensors are employed to acquire local information regarding risks and threats, such as congestion caused by evacuees, smoke, or harmful gases produced by fire or hazard levels owing to collapsed pathways. Interface devices such as smartphones, personal digital assistants (PDAs), and digital signage are used to advise evacuees about the paths they should follow.

A dynamic evacuation guidance system must function effectively in harsh environments that can damage its components, which is challenging. Therefore, it is unrealistic to expect that all system components will function as intended. For example, the devices, information lines, or sensors may fail, become disconnected, or generate inaccurate information, respectively. Thus, the system design should allow it to function despite damaged components. Although the system may not perform perfectly under these conditions, a single point of failure that can cause the entire system to malfunction should not exist.

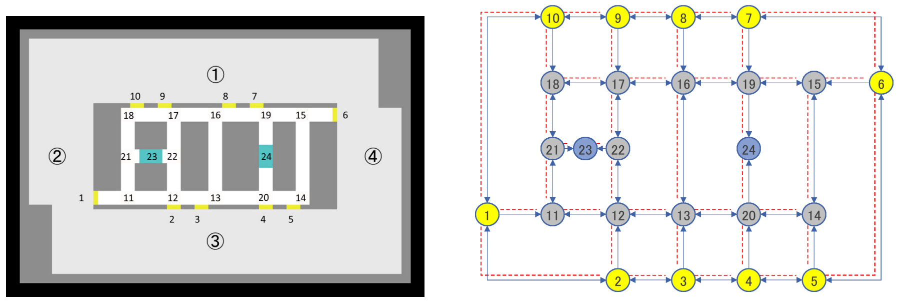
Thus, it is important to design dynamic evacuation guidance systems in a decentralized manner to ensure effective functionality under harsh conditions, which can prevent a single component or failure from causing the entire system to malfunction. This type of system typically has the following two primary objectives: 1) reducing the total evacuation time by mitigating the congestion caused by crowds, and 2) adapting evacuation routes to the dynamic environment. In this study, we present a distributed algorithm that uses only local information for effectively guiding evacuees in indoor situations. Assuming that all components operate without malfunctions, we focused on how a distributed system equipped with our algorithm can efficiently guide crowds to achieve these objectives. We did not consider the scenarios in which certain components of the system may malfunction.

## 2 RELATED WORK

The intelligent active dynamic signage system (IADSS) was developed as part of the GETAWAY project to overcome the many shortcomings of the

[a] https://orcid.org/0000-0003-2711-297X

Tsurushima, A.
Simulation Analysis of Evacuation Guidance Using Dynamic Distributed Signage.
DOI: 10.5220/0012268100003636
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Conference on Agents and Artificial Intelligence (ICAART 2024) - Volume 1*, pages 179-188
ISBN: 978-989-758-680-4; ISSN: 2184-433X
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.



(A) (B)

Figure 1: (A) Example floor plan. (B) Graphical representation of the floor plan.

widely used static signage systems for crowd evacuation guidance. IADSS was tested at the Sant Cugat Station in Barcelona, Spain with 1356 volunteers, demonstrating its effectiveness in crowd evacuation (Galea et al., 2014; Galea et al., 2017). The success of this project gained the attention of several researchers, and numerous studies have been conducted on dynamic evacuation signage systems. Lin et al. investigated the effectiveness of IADSS on more complex structures by using a fire dynamics simulator (Lin et al., 2017).

While IADSS was designed to draw the attention of the evacuees to signage information, studies have focused on the dynamic pathfinding of evacuees. Researchers have proposed dynamic route selection techniques, such as a combination of graph-based route representation, simulation-based evaluation techniques (Cisek and Kapalka, 2014), and density-based route selection techniques (Bernardini et al., 2016).

Subsequently, numerous optimization techniques have been developed for dynamic pathfinding evacuation, such as the combination of crowd modeling and artificial immune system-based route optimization (Khalid and Yusof, 2018), shortest pathfinding using the Dijkstra algorithm (Baidal et al., 2020), and a two-layer approach of dynamic sign assignment and pedestrian assignment (Li et al., 2022). Xue et. al. proposed a machine-learning technique using reinforcement learning (DQN) and evaluated its efficiency against the Dijkstra algorithm (Xue et al., 2021).

Novel approaches have been proposed for implementing dynamic evacuation guidance by using distributed systems. A congestion-awareness route selection approach using the cell phone of an evacuee was proposed (Kasai et al., 2017). A distributed technique using the Bellman-Ford dual subgradient algorithm was developed to guide groups of evacuees to use their cell phones (Zu and Dai, 2017). Nguyen et. al. developed an evacuation guidance system that divides a large building into several sections, calculates the optimal routes for each section, and subsequently generates the overall evacuation routes by coordinating these local sub-routes (Nguyen et al., 2022). A distributed architecture for real-time evacuation guidance that considers envy-freeness was proposed (Lujak et al., 2017). These studies generally focused on reducing the complexity of computing the global optimum and achieving system scalability.

In a pioneering study, Zhao et al. proposed that each node of the distributed evacuation guidance system independently calculates its direction of evacuation and shares this information with its neighboring nodes to develop the overall evacuation routes (Zhao et al., 2022). The system does not contain a single point of failure that controls the entire system, thus ensuring that it will continue to function despite certain components failing, which is expected in harsh environments during disaster evacuation. However, a detailed algorithm and its analysis were not clearly presented in their paper, and the performance of the distributed system was evaluated only in terms of fire detection time against a theoretical worst-case scenario.

In this study, we formulated a distributed evacuation guidance problem and proposed a distributed algorithm to generate the overall evacuation route. We also evaluated the performance of the distributed system in terms of congestion mitigation and adaptation to environmental changes using multi-agent simulations.

## 3 PROBLEM

Fig. 1(A) illustrates the evacuation environment (Tsurushima, 2022a) with the coordinate $(x,y) \in \mathbb{R}_1 \times \mathbb{R}_2$ where $\mathbb{R}_1 = [-39,41], \mathbb{R}_2 = [-26,24]$, which represents the positions in the environment used in this study. The environment consisted of a central core (dark gray square), with exits (depicted in blue with numbers), aisles (white), corners (depicted by numbers), and the surrounding space (light gray). Doors connecting the surrounding space and aisles in the central core are depicted in yellow, which evacuees in the surrounding space can use to move to the aisles and evacuate to the exits. Evacuation signs, which can advise evacuees regarding the directions for their next moves, are positioned in front of doors $(1,\ldots,10)$, at corners $(11,\ldots,22)$, and at two exits (23 and 24). Although not clearly illustrated, the environment contained sensors that indicate the levels of danger, such as congestion, smoke, or toxic gases produced by fire in the aisles and spaces near the doors.

Fig. 1(B) presents a graphical representation of Fig. 1(A). The nodes represent the doors, corners, and exits. The edges represent the corresponding aisles or regions in the surrounding space. The nodes in the figure also represent the evacuation signs positioned at the corresponding doors, corners, or exits. These signs and their positions can be interpreted interchangeably. The red dotted lines represent the communication channels between the two evacuation signs. The topologies of the aisles and device connections in the example are the same, which is not required in general cases. The blue arrows in the figure represent the directions of the next moves of the evacuees or the directions displayed on evacuation signs. For example, at door 1, an evacuee has three direction choices: door 2, door 10, and corner 11; the evacuation sign on door 1 indicates either one of doors 2, 10, or 11. The danger level in each region can be represented by the weights of the corresponding edges in the graph.

Each evacuation sign, which is a computational process in the system, acquires danger levels on the edges connecting the adjacent nodes using sensors, delivers data through communication channels, and displays the evacuation direction to adjacent nodes. Every evacuation sign functions autonomously and independently.

In this study, given the floor layout illustrated in Fig. 1, we investigated the generation of efficient evacuation plans for a dynamic evacuation environment using independently functioning evacuation signs. The study had two objectives: 1) to reduce the total evacuation time by mitigating the congestion caused by evacuees and 2) adapt to evacuation routes by considering dynamic environmental hazard levels.

## 3.1 Formulation

Consider the graph $\bar{G} = (V, \bar{E}, W)$ depicted in Fig. 1(B) with red dotted lines. Nodes $V = \{1, 2, \ldots, 24\}$ represent evacuation signs, undirected edges $\{i, j\} \in \bar{E}$ represent the communication channel between evacuation signs, and ordered lists $W = (w_1, \ldots, w_k, \ldots)$ represent the weight values $w_{I(i,j)}$ given by the corresponding sensors, where $I : V \times V \to \mathbb{N}$ maps $i, j$ to index $k$ in an ordered list. The weighted values represent the cost or level of danger at the edge if the evacuee uses an edge as part of the evacuation route; an evacuee should choose an evacuation route with a lower cost. $w_k$ and $w_{\{i,j\}}$ can be expressed interchangeably only if there is no confusion. Furthermore, $w_{\{ij\}} > 0$ and $w_{\{i,j\}} = w_{\{j,i\}}$.

In this study, we made the following assumptions:

1. $\bar{G}$ is a connected graph.
2. Any node $i \in V$ can communicate with the adjacent nodes $j \in \{j \in V | \{i, j\} \in \bar{E}\}$ and the sensors immediately associate with $w_{\{i,j\}}$.
3. All evacuation signs, sensors, and communication channels function without malfunctions.

1) indicates that any two nodes $i$ and $j$ on $\bar{G}$ have at least one path; $i, \ldots, j$ is denoted by $(i, j)$. 2) shows that $i$ can communicate with the adjacent node $\acute{i}$ on the path $(i, j)$, for example, $i, \acute{i}, \ldots, j$, which implies that $i$ can communicate with any node on path $(i, j)$ if $i$ is replaced with $\acute{i}$. Node $i$ can also communicate with any sensor because all the sensors on $\bar{G}$ are connected to at least one node on $\bar{G}$ and we assume that all the devices operate without malfunctions from 3). Accordingly, we make the following assumption:

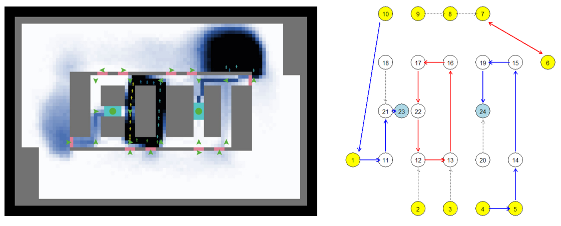
**Assumption 1.** *Any node $i$ on $\bar{G}$ can instantly communicate with any other node $j$ and sensor $w_{\{i,j\}}$ on $\bar{G}$.*

Assumption 1 does not hold for real evacuation situations; however, in this paper, we discuss all the technical details under this assumption for simplicity.

We now consider a directed graph $G = (V, E, W)$, as shown in Fig. 1(B), with blue arrows, where $e_k \in E$ is the directed edge $e_k = (i, j)$, which represents a set of adjacent nodes $j$, where evacuation sign $i$ can point out or an evacuee at $i$ can select the next move. Finally, we introduce a set of exits, $V_g = \{23, 24\}$.

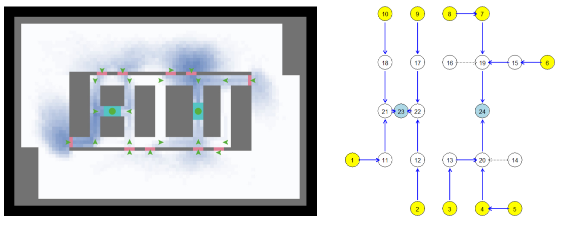
Given the dynamic nature of the problem stated in this section, namely $E$ and $W$ continuously vary and may need to be denoted temporally, for example, as $E(t)$ and $W(t)$. However, for simplicity, $E$ and $W$ are considered and treated as static information until Section 5. The dynamic nature of the problem is examined in Section 6; both $E$ and $W$ are treated as dynamic information in this section.

**Definition 1.** *We call $G$ a universal evacuation graph. The set of nodes on $G$ being pointed out by $i$, which indicates any possible direction that an evacuee in $i$ can choose, is denoted by $\Delta : V \to 2^V$, $j \in \Delta(i)$, $(i, j) \in E$.*



(A) (B)

Figure 2: (A) Evacuation graph with cycles. (B) Graph representation of evacuation graph with cycles.



(A) (B)

Figure 3: (A) Evacuation graph without cycles. (B) Graph representation of evacuation graph.

**Definition 2.** *We call $\dot{G} = (V, \dot{E}, \dot{W})$ a evacuation graph, where $\dot{E} = \{(i, j) \in E\}$ denotes the set of edges when an evacuation sign $i$ points to an adjacent evacuation sign $j$ and a set of corresponding edge weights $\dot{W}$. The evacuation sign indicated by $i$ on $\dot{G}$ is denoted by $\delta : V \to V$, $(i, \delta(i)) \in \dot{E}$.*

Note that $\delta(i) \in \Delta(i)$.

**Corollary 1.** *Given evacuation graph $\dot{G}$, $\dot{E} \subset E$, $\dot{W} \subset W$, where $|V| - |V_g| = |\dot{E}| = |\dot{W}|$ and $\forall i\ \exists j\ (i \in \{V \setminus V_g\},\ (i, j) \in \dot{E})$.*

A evacuation graph is a graph in which each evacuation sign $i$ advises evacuees to a specific evacuation route by pointing to an adjacent evacuation sign (node); each evacuation sign points to only one node. Now, we consider the special case of evacuation graphs.

## 3.2 Cycle

**Definition 3.** *We denote $\ddot{G} = (V, \ddot{E}, \ddot{W})$, where $\ddot{E} = \{(i, j) \in \dot{E} \mid \dot{E}$ is acyclic$\}$, an efficient evacuation graph.*

If we denote a tree by $\mathcal{T} = (V, E, W)$, the following lemma holds.

**Lemma 1.** *An efficient evacuation graph can be represented as $\ddot{G} = \{\mathcal{T}_1, \ldots, \mathcal{T}_{|V_g|}\}$, where $\mathcal{T}_i$ is a tree with element $V_g$ as its root.*

**Proof.** We divide $\ddot{G} = (V, \ddot{E}, \ddot{W})$ into $\ddot{G} = \{G_1, \ldots, G_{|V_g|}\}$, $V = \{V_1, \ldots, V_{|V_g|}\}$, $\ddot{E} = \{\ddot{E}_1, \ldots, \ddot{E}_{|V_g|}\}$, where $\ddot{E}_k = \{(i, j) \in \ddot{E} \mid i, j \in V_k\}$, $\nexists (i, j)\ (i \in V_m,\ j \in V_n,\ m \neq n)$ ($\ddot{W}$ is divided in the same manner as $\ddot{E}$). Corollary 1 indicates that $|V| - |V_g| = |V_1| + \ldots + |V_{|V_g|}| - |V_g| = |V_1| - 1 + \ldots + |V_{|V_g|}| - 1 = |\ddot{E}_1| + \ldots + |\ddot{E}_{|V_g|}| = |\ddot{E}|$. Now, we have $|\ddot{E}_k| = |V_k| - 1$ because $|V_k|$ has only one element of $|V_g|$ and $\ddot{E}_k$ does not have a cycle (Definition 3). Therefore, $G_k = \mathcal{T}_k = (V_k, \ddot{E}_k, \ddot{W}_k)$ represents a tree with element $V_g$ as a root. □

In this section, we discuss how the evacuation graphs with and without cycles affect the evacuation guidance results using two examples.

Fig. 2 (B) presents an evacuation graph with two cycles: 12, 13, 16, 17, 22 and 6, 7 illustrated by red arrows; blue arrows indicate evacuation routes from doors to exits. Fig. 2 (A) presents the assignment of evacuation directions displayed by each evacuation sign associated with evacuation graph (B) at each location in the floor plan. The dark blue regions in (A) indicate the regions where traffic jams are caused by the evacuees when the evacuation simulations were performed using evacuation graph (B). By comparing these two figures, we observed that the congestion of the evacuees occurs when the cycles are formed in the evacuation graph.

Fig. 3 (B) depicts an evacuation graph without cycles (efficient evacuation graph). This graph has no red arrows and consists of two trees whose roots are exits (Lemma 1). Fig. 3 (A) presents the evacuation simulation results in the floor plan corresponding to efficient evacuation graph (B). There are no dark blue areas in this figure, indicating that evacuees did not cause traffic jams.

According to the simulation results, severe congestion can occur when the evacuation graph includes cycles. Therefore,

**Requirement 1.** *the formation of cycles in the evacuation graph should be avoided in evacuation guidance,*

despite the individual evacuation signs independently generating the evacuation routes using only local information.

In addition, the cycles on the evacuation graph do not lead the evacuees to any exits, giving them the impression that the system is inconsistent and resulting in a loss of confidence in the system. This is another reason the cycles on the evacuation graph should be avoided.

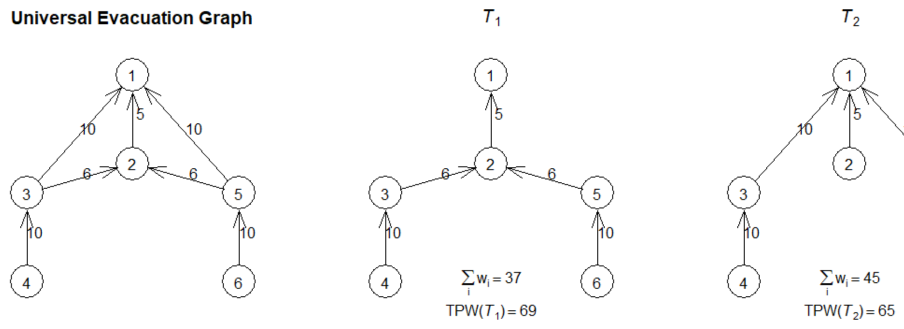


Figure 4: Universal evacuation graph (left). Minimum spanning tree (center). Minimum efficient evacuation graph (right).

## 3.3 Edge Weight

Considering the universal evacuation graph presented on the left in Fig. 4, suppose that node 1 is the exit and the numbers on the edges are the weight values representing the costs or levels of danger in evacuation routes, such as congestion, smoke, or toxic gases. A route with a lower weight was preferred. $\mathcal{T}_1$ (the center of Fig. 4) is the minimum spanning tree of the universal evacuation graph, whose weight is $\sum_i w_i = 37$. $\mathcal{T}_2$ (right-hand side of Fig. 4) is another spanning tree of the universal evacuation graph whose weight is $\sum_i w_i = 45$, which is not the minimum. Let us denote the path from $i$ to $j$ on tree $\mathcal{T}(i,j)$, and the total weights of the edges on the path $weight(\mathcal{T}(i,j))$. Subsequently, $\forall i\ weight(\mathcal{T}_1(i,1)) \geq weight(\mathcal{T}_2(i,1))$. For example, $weight(\mathcal{T}_1(4,1)) = 21$ and $weight(\mathcal{T}_2(4,1)) = 20$; the latter is smaller than the values of the former. Thus, to implement an efficient evacuation guidance we must determine evacuation routes consisting of trees with minimum route weights, as shown for $\mathcal{T}_2$ in Fig. 4.

We define the total path weight ($TPW$) of tree $\mathcal{T}$ as follows:

**Definition 4.** *$TPW(\mathcal{T}) = \sum_{i \in V \setminus \{g\}} weight(\mathcal{T}(i,g))$, where $g$ denotes the exit node of $\mathcal{T}$.*

We also define $TPW$ of the efficient evacuation graph $\ddot{G} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots\}$ as follows:

**Definition 5.** $TPW(\ddot{G}) = \sum_j TPW(\mathcal{T}_j)$.

**Definition 6.** *We call an efficient evacuation graph with a minimum TPW a minimum efficient evacuation graph $G^*$.*

Now, we have an evacuation guidance problem, that is, $\min TPW(\ddot{G})$ s.t. $\ddot{G} \in \ddot{\mathcal{G}}$, where $\ddot{\mathcal{G}}$ is a set of efficient evacuation graphs. To implement an efficient evacuation guidance

**Requirement 2.** *we must find the minimum efficient evacuation graph on a universal evacuation graph.*

Algorithm 1: Broadcast.

```
Local Variable: $\widehat{E}, \widehat{W}, \widehat{T}$
1  Function
   Broadcast($o$, $i$, $\delta(o)$, $t$, $\Omega_W$, $\Upsilon_W$):
2      if $\widehat{T}[o] < t$ then
3          $\widehat{T}[o] \leftarrow t$;
4          $\widehat{E}[o] \leftarrow \delta(o)$;
5          foreach $j \in 1, \ldots, |\Omega_W|$ do
6              $\widehat{W}[\Omega_W[j]] \leftarrow \Upsilon_W[j]$
7          end
8          foreach $a \in \Delta(i)$ do
9              Call
               Broadcast($o$, $a$, $\delta(o)$, $t$, $\Omega_W$, $\Upsilon_W$)
               on node $a$
10         end
11     end
12 end
```

## 4 DISTRIBUTED ALGORITHM

Evacuation guidance systems must operate under extreme conditions where not all components are expected to function as intended;

**Requirement 3.** *the system should be resilient, have no single point of failure, and be distributed.*

In this section, we present a distributed algorithm satisfying Requirement 1 – 3 under Assumption 1. Each universal evacuation graph node is equipped with the algorithm that repeatedly executes at certain intervals; the algorithm is distributed but not fully asynchronous. The proposed algorithm consists of two steps: 1) *Broadcast*, which propagates local information throughout the graph, and 2) *UpdateSign*, which determines the optimal evacuation routes based on the information provided by *Broadcast*.

### 4.1 Broadcast

*Broadcast* collects local information, including the current pointing direction of node $i$ and the edge weights obtained from the connected sensors, and distributes them to all the adjacent nodes. Each node repeatedly broadcasts this information throughout the graph, and all the other nodes acquire them to construct a local image of the global graph. We assume that the times at which the nodes broadcast vary but all the nodes finish broadcasting within a certain period. At this point, all the nodes have local images of the global graph $\widehat{G} = (V, \hat{\dot{E}}, \widehat{W})$. We assume that each node has local variables $\hat{\dot{E}}, \widehat{W}, \widehat{T}$, and $\Delta i$. $\widehat{T}$ is an ordered list containing the last broadcast reception times of all the nodes, and $\widehat{T}[i]$ presents the $i$th element of the ordered list $\widehat{T}$. It also has two other ordered lists: $\Omega_W$ to hold the edge indices of $\widehat{W}$ and 2) $\Upsilon_W$ to maintain their corresponding values, which contain the local edge weights collected by node $i$.

*Broadcast* at node $i$ is presented in Algorithm 1, where the originator node is indicated by $o$, the node id is $i$, the current direction of the originator is $\delta(o)$, and the time at which the originator sent the broadcast is $t$. *Broadcast* is a simple flooding algorithm that requires $2|\bar{E}||V| - |V|^2 + |V|$ communications to update the given value of universal evacuation graph.

### 4.2 UpdateSign

Algorithm 2: UpdateSign.

```
   Local Variable: $\hat{\dot{E}}, \widehat{W}, \delta(i)$
1  Function UpdateSign($i$):
2      $l^*_{ig} \leftarrow$ Search($i$, $\widehat{W}$);
3      ($i$, $next$) $\leftarrow l^*_{ig}[1]$;
4      $\hat{\dot{E}}[i] \leftarrow (i, next)$;
5      if $\hat{\dot{E}}$ include a cycle then
6          call UpdateSign($next$) on $next$;
7          await $\hat{\dot{E}} \leftarrow$ UpdateSign($next$) from $next$
8      end
9      $\delta(i) \leftarrow \hat{\dot{E}}[i]$;
10     return $\hat{\dot{E}}$
11 end
```

Given a set of weights $\widehat{W}(t)$ at any point in time $t$, our algorithm generated a series of efficient evacuation graphs and converged to the minimum efficient evacuation graph as below.

$$\ddot{G}_1 \rightarrow \ddot{G}_2 \rightarrow \ldots \rightarrow G^* \text{ where } TPW(\ddot{G}_n) \geq TPW(\ddot{G}_{n+1})$$

$\ddot{G}_n \rightarrow \ddot{G}_{n+1}$ demonstrates that by executing the algorithm, node $i$ in $\ddot{G}_n$ pointing to node $j$ changes its pointing direction to another node $\acute{j}$, resulting in the transformation of $\ddot{G}_n$ into $\ddot{G}_{n+1}$, which is also an efficient evacuation graph.

In reality, each node executes its algorithm at different times and the local image varies from node-to-node because $\widehat{W}$ represents an evacuation environment, which is dynamic. *UpdateSign* at each node searches for the evacuation routes with a minimum *TPW* for different graphs, resulting in inconsistent evacuation routes which may include cycles. Therefore, *UpdateSign* requires a mechanism to break the cycles when it finds cycles on the evacuation graph.

Subsequently, we consider the case in which node $i \in \mathcal{T}_k$ on $\ddot{G} = \{\mathcal{T}_k, \mathcal{T}_l\}$ (Lemma 1) changes its pointing direction to the other nodes step-by-step. Edge $(i, j)$ is removed from $\ddot{G}$ (STEP 1) and a new edge $(i, \acute{j})$ is added to the resulting graph (STEP 2).

After STEP 1, because the tree has minimal connectivity, $\mathcal{T}_k$ is split into two trees: $\mathcal{T}_k^P$, which excludes node $i$, and $\mathcal{T}_k^C$, which includes node $i$ as its root.

**Theorem 1.** *A graph consisting of three trees was obtained.* $\tilde{\ddot{G}} = \{\mathcal{T}_k^P, \mathcal{T}_k^C, \mathcal{T}_l\}$ *after STEP 1.*

By adding a new edge $(i, \acute{j})$ to $\tilde{\ddot{G}}$ in STEP 2, the following theorem holds:

**Theorem 2.** *When node i on $\mathcal{T}_k$ changes its pointing direction from j to $\acute{j}$, the resulting graph $\ddot{G}_{new}$ is an efficient evacuation graph only if $\acute{j} \in \mathcal{T}_l$ and $\acute{j} \in \mathcal{T}_k^P$.*

**Proof.** In these three cases, a new edge $(i, \acute{j})$ is added to $\tilde{\ddot{G}} = \{\mathcal{T}_k^P, \mathcal{T}_k^C, \mathcal{T}_l\}$ (Theorem 1).
1. $\acute{j} \in \mathcal{T}_l$. Both $\mathcal{T}_k^C$ and $\mathcal{T}_l$ are trees, and the resulting graph $\acute{\mathcal{T}_l}$ created by adding an edge from the root of $\mathcal{T}_k^C$ to any node of $\mathcal{T}_l$ is also a tree. Therefore, $\ddot{G}_{new} = \{\mathcal{T}_k^P, \acute{\mathcal{T}_l}\}$ is an efficient evacuation graph.
2. $\acute{j} \in \mathcal{T}_k^P$. Both $\mathcal{T}_k^C$ and $\mathcal{T}_k^P$ are trees and the resulting graph $\acute{\mathcal{T}_k}$ created by adding an edge $(i, \acute{j})$ is also a tree. Therefore, $\ddot{G}_{new} = \{\acute{\mathcal{T}_k}, \mathcal{T}_l\}$ is an efficient evacuation graph.
3. $\acute{j} \in \mathcal{T}_k^C$. $\acute{j}$ is any node on $\mathcal{T}_k^C$ except $i$. One path exists between $i$ to $\acute{j}$ on $\mathcal{T}_k^C$ because it is a tree. Thus, adding a new edge $(i, \acute{j})$ will form a cycle on $\mathcal{T}_k^C$. Therefore, $G_{new} \in \{\dot{G} \setminus \ddot{G}\}$ is *not* an efficient evacuation graph. □

Let $l_{ij} = ((i, i_1), (i_1, i_2), \ldots, (i_{n-1}, j))$ be a path from node $i$ to node $j$ in graph $G$. The weight of path $l_{ij}$ is denoted by $weight(l_{ij})$, given by $weight(l_{ij}) = \sum_{(i,j) \in l_{ij}} w_{(ij)}$, where $w_{(ij)} \in W$ represents the weights associated with the edges in $W$. The path with the minimum weight is denoted as $l^*_{ij}$, and its weight as $w^*_{(ij)} = weight(l^*_{ij})$. Let $l^*_{ig}$, $g \in V_g$ represent the path with the minimum *TPW* from node $i$ to one of the exits on graph $G$, and suppose we have a function $Search(i, \widehat{W})$ that determines $l^*_{ig}$ in graph $G$ with $\widehat{W}$.

The distributed algorithm *UpdateSign* at node $i$ that converges to $G^*$ is given by Algorithm 2. **Local Variable:** in Algorithm 2 presents the local variables available at node $i$. Node $i$ executes $UpdateSign(i)$ on itself when it starts the algorithm, and $\widehat{E}$ returned by the algorithm contains the updated routes with a minimum *TPW*.

## 5 ANALYSIS

In this section, an analysis of the proposed algorithm is presented under Assumption 1 assuming that $W$ remains unchanged.

**Theorem 3.** *UpdateSign stops after a finite number of steps.*

**Proof.** Under Assupmtion 1, *UpdateSign* will only stop if the *next* node in line 7 of Algorithm 2 does not return. Suppose *Search* in line 2 finds path $l^*_{ig1}$ with a minimum *TPW* from node $i$ to exit $g1$. This occurs when a certain node $j$ on $l^*_{ig1}$ recursively calls node $i$ in line 6 of *UpdateSign* . There are two possible scenarios. First, node $j$ finds the path from $j$ to $g1$ such that $l^*_{jg1} = (j, \ldots, i, \ldots, g1)$. $w^*_{(j,g1)} = w^*_{(j,i)} + w^*_{(i,g1)}$ because $w^*_{(j,g1)}$ is a minimum. However, $w^*_{(i,g1)} = w^*_{(i,j)} + w^*_{(j,g1)} = w^*_{(i,j)} + w^*_{(j,i)} + w^*_{(i,g1)}$, which leads to $w^*_{(i,g1)} = 2w^*_{(i,j)} + w^*_{(i,g1)}$ because $w_{(i,j)} = w_{(j,i)}$. This contrasts with the definition $w_{(i,j)} > 0$. Second, node $j$ finds the path with a minimum *TPW* from $j$ to $g2$ such that $l^*_{ig2} = (j, \ldots, i, \ldots, g2)$, where $g2$ is the other exit. Because $l^*_{ig1} = (i, \ldots, j, \ldots, g1)$ has the minimum *TPW* , $w^*_{(i,g1)} = w^*_{(i,j)} + w^*_{(j,g1)} \leq w^*_{(i,g2)}$. However, node $j$ also has path $l^*_{jg2}$ with a minimum *TPW* $w^*_{(j,g2)}$ which includes node $i$; therefore, $w^*_{(j,g2)} = w^*_{(j,i)} + w_{(i,g2)} \leq w^*_{(j,g1)}$. Now, $w^*_{(i,j)} + w^*_{(j,g1)} \leq w^*_{(j,g1)} - w^*_{(j,i)}$, which leads to $w^*_{(i,j)} + w^*_{(j,i)} \leq 0$. This is contrary to the definition $w_{(i,j)} > 0$. Thus, line 7 of Algorithm 2 must return. □

**Theorem 4.** *UpdateSign always generates efficient evacuation graphs.*

**Proof.** Theorem 2 indicates that $\dot{E}$ is not an efficient evacuation graph only if node $i$ points to node $j$ such that $j \in \mathcal{T}^C_k$. However, in this case, line 5 of Algorithm 2 ensures that $\dot{E}$ passes recursively to *UpdateSign* on the adjacent node. Theorem 3 indicates that this recursive call stops after a finite number of steps, and $\ddot{E}$ returns to line 7 in Algorithm 2 and does not include $l^*_{ig}$ that recursively contains $i$. Thus, *UpdateSign* always generates an efficient evacuation graph. □

Consider a system $S$ in which each evacuation sign has an *UpdateSign* and executes it randomly.

**Theorem 5.** *System S generates a series converging on minimum efficient evacuation graph such that*

$$\ddot{G}_1 \to \ddot{G}_2 \to \ldots \to G^* where\, TPW(\ddot{G}_n) \geq TPW(\ddot{G}_{n+1})$$

**Proof.** Consider that node $i$ on $\mathcal{T}_{old}$ on $\ddot{G}$ executes *UpdateSign* . We have $\ddot{G} = \{\mathcal{T}^P_{old}, \mathcal{T}^C_{old}, \mathcal{T}_l\}$ (Theorem 1) and let $\mathcal{T}_{new}$ be a new tree having node $i$ after executing *UpdateSign* . $\mathcal{T}_{new}$ is generated using the method given in Cases 1 and 2 in the proof of Theorem 2. First, $\acute{j} \in \mathcal{T}_l$. Let the exits before and after *UpdateSign* be $g$ and $\acute{g}$, respectively, which differ from one another. $TPW(\mathcal{T}_{old}(i,g)) \geq TPW(\mathcal{T}_{new}(i,\acute{g}))$ because *Search* in line 2 of *UpdateSign* determines the path with the minimum *TPW* . As both $\mathcal{T}^P_{old}$ and $\mathcal{T}_l$ remain unchanged, and $\mathcal{T}^C_{old}$ is a tree with root $i$, the following relationships hold for all the nodes $k$ in $\mathcal{T}^C_{old}$ except $i$:

$$\begin{aligned} &TPW(\mathcal{T}_{new}(k,\acute{g})) \\ &\quad = TPW(\mathcal{T}_{new}(k,i)) + TPW(\mathcal{T}_{new}(i,\acute{g})) \\ &\quad \leq TPW(\mathcal{T}_{new}(k,i)) + TPW(\mathcal{T}_{old}(i,g)) \\ &\quad = TPW(\mathcal{T}^C_{old}(k,i)) + TPW(\mathcal{T}_{old}(i,g)) \\ &\quad = TPW(\mathcal{T}_{old}(k,g)). \end{aligned}$$

Second, $\acute{j} \in \mathcal{T}^P_{old}$. In this case, exit $g$ remains unchanged. $TPW(\mathcal{T}_{old}(i,g)) \geq TPW(\mathcal{T}_{new}(i,g))$, and $\mathcal{T}^P_{old}$ and $\mathcal{T}_l$ also remain unchanged, yielding in the same manner $TPW(\mathcal{T}_{old}(j,g)) \geq TPW(\mathcal{T}_{new}(j,g))$. Therefore, $TPW(\ddot{G}_{old}) \geq TPW(\ddot{G}_{new})$. Furthermore, Theorem 4 yields that *UpdateSign* always generates efficient evacuation graph, and the series above is obtained. □

**Theorem 6.** *UpdateSign is self-stabilizing.*

**Proof.** Suppose that system $S$ begins from an initial graph with cycles $G_{init} \in \{\dot{G} \setminus \ddot{G}\}$. At a certain point, node $i$ executes *UpdateSign* in a cycle. *Search* in line 2 of Algorithm 2 will find a path to an exit $l^*_{ig}$; subsequently, line 5 of the algorithm finds where $i$ denotes the cycle. It recursively calls *UpdateSign* on the next node of $l^*_{ig}$ in line 6. However, this recursive call terminates in a finite number of steps according to Theorem 3. At this point, a cycle must be broken owing

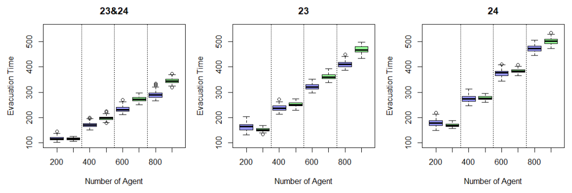


Figure 5: Mitigating Congestions.

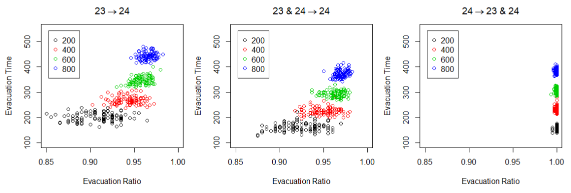


Figure 6: Adapting the Evacuation Route.

Table 1: Mean evacuation time in Fig. 5.

| Scenario | | 200 | 400 | 600 | 800 |
|---|---|---|---|---|---|
| 23&24 | dynamic | 117.8 | 172.0 | 233.6 | 290.6 |
| | static | 116.2 | 198.1 | 272.7 | 345.5 |
| 23 | dynamic | 164.3 | 238.0 | 322.3 | 410.9 |
| | static | 152.2 | 251.5 | 361.4 | 468.5 |
| 24 | dynamic | 178.8 | 275.3 | 377.0 | 473.0 |
| | static | 170.4 | 277.4 | 384.4 | 501.2 |

Table 2: Mean evacuation time and ratio in Fig. 6.

| Scenario | | 200 | 400 | 600 | 800 |
|---|---|---|---|---|---|
| 23 | ratio | 0.91 | 0.94 | 0.96 | 0.97 |
| | time | 201.7 | 267.7 | 346.7 | 443.8 |
| 23&24 | ratio | 0.92 | 0.95 | 0.96 | 0.97 |
| | time | 162.1 | 225.2 | 293.5 | 372.8 |
| 24 | ratio | 1.00 | 1.00 | 1.00 | 1.00 |
| | time | 157.1 | 234.6 | 306.1 | 383.4 |

to Theorem 4; a series from $G_{init}$ to $\ddot{G}$ is formed. Theorem 5 ensures that the series never forms a cycle and generates only efficient evacuation graphs. Thus, *UpdateSign* is self-stabilizing. $\square$

The evacuation environment is constantly changing and evacuees always move to different locations; thus, $W$ is constantly changing information and it is usually not possible to obtain a minimum efficient evacuation graph. Despite considering these facts, Theorem 4 ensures that *UpdateSign* always generates efficient evacuation graphs.

**Corollary 2.** *System $S$ generates a series of efficient evacuation graphs, despite $W$ being dynamical.*

## 6 EXPERIMENT

Thus far, we have treated $E$ and $W$ as static information for simplicity, whereas in real situations they are dynamic. Both the evacuees and evacuation signs are autonomous entities that affect one another, complicating the prediction of the overall behavior of the system, for example, the evacuation may never end owing to oscillations caused by these two entities. In this section, we treat $E$ and $W$ as dynamic information and evaluate the performance of the dynamic distributed signage system using multi-agent simulations (Wilensky, 1999) assimilating real evacuation situations.

### 6.1 Mitigating Congestion

Evacuation simulation experiments were conducted to investigate the effectiveness of the dynamic signage system in reducing congestion during evacuations. We highlighted these results by comparing them with those of a static signage system. To create congestion during evacuation, evacuation agents were initially placed in ① and ②, as shown in Fig. 1(A); ③ and ④ were left empty. We also assumed $w_{\{i,j\}} = d(n-1) > 0$, where $d$ is the length of the edge $\{i, j\}$ and $n$ is the number of agents currently on $\{i, j\}$.

To illustrate the effects of the dynamic signage system, we use an agent with a simple random choice model as follows:

1. When the agent is in the surrounding space, it will randomly select doors that are visible from its location, for example, if the agent is at ①, it will randomly select doors 7–10.
2. When the agent is at a corner in the central core, it will randomly select a direction, as indicated by the blue arrows in Fig. 1(B), for example, if the agent is at corner 12, it will randomly select directions 11, 13, or 22.
3. If the agent is in the central core and there is an evacuation sign in its field of view, the agent will follow the evacuation sign rather than make a random selection.

The results of the three scenarios, including both available, Exit 23 available, and Exit 24 available, are shown in the left, center, and right diagrams in Fig. 5, respectively, and are summarized in Table 1. In this figure, the x-axis indicates the number of evacuation agents, that is, 200, 400, 600, and 800; the y-axis represents the evacuation time. A total of 100 simulations were conducted for each case. We plotted the results for the static signage system in green, and the dynamic signage system in blue.

## 6.2 Adapting the Evacuation Route

We also conducted experiments to investigate how the dynamic signage system adapts evacuation routes to dynamic environments. In this setting, Exit 23 was initially the only available exit; however, the available exit changed to Exit 24 when half of the evacuees exited. The evacuation graph possessed by each node changes immediately at this point. We measured the ratio of agents who evacuated via an available exit; the agents evacuated via Exit 23 in the first half of the simulation and via Exit 24 in the second half. The following two other variations of the scenario were examined: 1) Both exits were initially available, followed by Exit 24 being the only available exit. 2) Exit 24 was initially the only exit available, followed by both exits becoming available .

The results of the three scenarios are presented in Fig. 6. The results of the scenarios in which Exits 23, 23&24, and 24 were initially available are shown in the left, center, and right diagrams of Fig. 6 and the means of these results are summarized in Table 2.

## 7 RESULT AND DISCUSSION

Fig. 5 reveals that by comparing the static (green) and dynamic (blue) evacuation signs, the latter generally provides a shorter evacuation time, as the blue plots are below the green plots. However, when the number of evacuation agents is small (e.g., 200), the static evacuation signs tend to provide shorter evacuation times compared with the dynamic evacuation signs, confirming the effectiveness of the dynamic evacuation signs when the number of evacuees is large and congestion was anticipated. This suggests that dynamic evacuation signs may incur certain costs in evacuation guidance, such as additional time owing to collisions and confusion among the evacuees, resulting in longer evacuation times.

Fig. 6 demonstrates that our algorithm could guide approximately 90% of evacuees to the intended exits, except when the number of evacuees was small (e.g., 200). The figure demonstrates that as the number of evacuees increases, the proportion of evacuees guided to the intended exits increases. The diagram on the right side of Fig. 6, where Exit 24 is first available followed by both Exits 23 and 24, indicates that nearly 100% of the evacuees were directed to the intended exit when an available exit was initially limited, after which all the exits became available. This indicates that unintended steering generally occurs immediately after the available exit changes, which is owing to the change in the evacuation graph possessed by each node. This is because it is easier to direct more evacuees to available exits if the available exit is initially restricted and the restriction is later removed, as shown in the right diagram in Fig 6. Theorem 6 also ensures that our algorithm can continue to generate efficient evacuation graphs despite the subsequent turbulence caused by the change in efficient evacuation graphs.

In this study, we proposed a distributed algorithm that provides efficient evacuation guidance to mitigate congestion during evacuation and adapts evacuation routes based on the changes that occur in evacuation environments. One limitation of this algorithm is that although it is decentralized, it is not fully asynchronous. In actual distributed systems, each device cannot predict whether the message it sends will be received by the other device within a certain time, or in the order in which they were sent. However, the device cannot predict that the task delegated to the other device is completed, and whether the results for the tasks are returned within a certain period. Line 7 of Algorithm 2 assumes that the *UpdateSign* on the adjacent node will be completed and its result will be returned within a certain time, which is not guaranteed in real situations, leading to the termination of the entire process. This is particularly critical if the system contains faulty devices that are unable to return responses. Therefore, it is crucial to make this algorithm asynchronous to address this problem and analyze how the system performs in faulty environments.

Other important aspects such as human factors were not investigated in this study. The human decision-making process is more complex, making evacuation behavior unpredictable, especially wayfinding during evacuations (Lovreglio et al., 2016; Vizzari et al., 2020; Andresen et al., 2018), while we employed a simple agent model in the simulations. Furthermore, the visual functions of evacuees affect their ability to find evacuation signs (Ding, 2020; Tsurushima, 2021; Tsurushima, 2022b). Whether people follow an evacuation sign when it rapidly changes directions is also critical but it has not been tested thus far. These topics should be investigated in future studies.

## 8 CONCLUSION

We proposed a distributed algorithm for a dynamic evacuation guidance system to mitigate congestion during evacuation and to advise evacuees on evacuation routes that are adaptable to environmental changes. This system has no central control mechanism, and the devices equipped with our algorithm only use local information and communicate with neighboring devices. Simulation results demonstrated that the proposed algorithm can provide evacuation guidance, resulting in a shorter evacuation time compared with static evacuation guidance; furthermore, approximately 90% of the evacuees were guided to the intended routes despite the available exits changing during evacuation.

## ACKNOWLEDGMENT

The author would like to thank Mr. Kei Marukawa for his assistance and helpful discussions.

## REFERENCES

Andresen, E., Chraibi, M., and Seyfried, A. (2018). A representation of partial spatial knowledge: a cognitive map approach for evacuation simulations. *Transportmetrica A: Transport Science*, 14(5-6):433–467.

Baidal, C., Arreaga, N., and Padilla, V. (2020). Design and testing of a dynamic reactive signage network towards fire emergency evacuations. *International Journal of Electrical and Computer Engineering (IJECE)*, 10:5853.

Bernardini, G., Azzolini, M., D'Orazio, M., and Quagliarini, E. (2016). Intelligent evacuation guidance systems for improving fire safety of Italian-style historical theatres without altering their architectural characteristics. *Journal of Cultural Heritage*, 22:1006–1018.

Cisek, M. and Kapalka, M. (2014). Evacuation route assessment model for optimization of evacuation in buildings with active dynamic signage system. *Transportation Research Procedia*, 2:541–549.

Ding, N. (2020). The effectiveness of evacuation signs in buildings based on eye tracking experiment. *Natural Hazards*, 103.

Galea, E., Xie, H., Deere, S., Cooney, D., and Filippidis, L. (2017). Evaluating the effectiveness of an improved active dynamic signage system using full scale evacuation trials. *Fire Safety Journal*, 91.

Galea, R. E., Xie, H., and Lawrence, J. P. (2014). Experimental and survey studies on the effectiveness of dynamic signage systems. *Fire Safety Science*, 11:1129–1143.

Kasai, Y., Sasabe, M., and Kasahara, S. (2017). Congestion-aware route selection in automatic evacuation guiding based on cooperation between evacuees and their mobile nodes. *EURASIP Journal on Wireless Communications and Networking*, 164.

Khalid, M. N. A. and Yusof, U. K. (2018). Dynamic crowd evacuation approach for the emergency route planning problem: Application to case studies. *Safety Science*, 102:263–274.

Li, M., Xu, C., Xu, Y., Ma, L., and Wei, Y. (2022). Dynamic sign guidance optimization for crowd evacuation considering flow equilibrium. *Journal of Advanced Transportation*, 2022:2555350.

Lin, H.-M., Chen, S.-H., Kao, J., Lee, Y.-M., Lin, C.-Y., and Hsiao, G. (2017). Applying active dynamic signage system in complex underground construction. *International Journal of Scientific & Engineering Research*, 8.

Lovreglio, R., Fonzone, A., and dell'Olio, L. (2016). A mixed logit model for predicting exit choice during building evacuations. *Transaportation Research Part A: Policy and Practice*, 92:59–75.

Lujak, M., Billhardt, H., Dunkel, J., Fernández, A., Hermoso, R., and Ossowski, S. (2017). A distributed architecture for real-time evacuation guidance in large smart buildings. *Computer Science and Information Systems*, 14:257–282.

Nguyen, V.-Q., Vu, H.-T., Nguyen, V.-H., and Kim, K. (2022). A smart evacuation guidance system for large buildings. *Electronics*, 11:2938.

Tsurushima, A. (2021). Simulation analysis of tunnel vision effect in crowd evacuation. In Rutkowski, L., Scherer, R., Korytkowski, M., Pedryca, W., Tadeusiewicz, R., and Zurada, J. M., editors, *Artificial Intelligence and Soft Computing. ICAISC 2021. Lecture Notes in Computer Science*, volume 12854, pages 506–518. Springer.

Tsurushima, A. (2022a). Efficient crowd evacuation guidance with multiple visual signage using a middle-range agent model and black-box optimization. In *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2591–2598.

Tsurushima, A. (2022b). Tunnel vision hypothesis: Cognitive factor affecting crowd evacuation decisions. *SN Computer Science*, 3(332).

Vizzari, G., Crociani, L., and Bandini, S. (2020). An agent-based model for plausible wayfinding in pedestrian simulation. *Engineering Applications of Artificial Intelligence*, 87:103241.

Wilensky, U. (1999). NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

Xue, Y., Wu, R., Liu, J., and Xianglong, T. (2021). Crowd evacuation guidance based on combined action-space deep reinforcement learning. *Algorithms*, 14(26).

Zhao, H., Schwabe, A., Schläfli, F., Thrash, T., Aguilar, L., Dubey, R., Karjalainen, J., Hölscher, C., Helbing, D., and Schinazi, V. (2022). Fire evacuation supported by centralized and decentralized visual guidance systems. *Safety Science*, 145:105451.

Zu, Y. and Dai, R. (2017). Distributed path planning for building evacuation guidance. *Cyber-Physical Systems*, 3(1-4):1–21.