Automation in Construction 59 (2015) 139–148

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

An automated direction setting algorithm for a smart exit sign

Jehyun Cho, Ghang Lee *, Seungjae Lee

Department of Architectural Engineering, Yonsei University, Seoul, South Korea

ARTICLE INFO

Article history:
Received 12 November 2014
Received in revised form 23 April 2015
Accepted 13 May 2015
Available online 27 May 2015

Keywords:
Shortest-path algorithm
Egress
Evacuation
Fire
Safety
Smart exit sign
Indoor navigation
Wayfinding

ABSTRACT

The aim of this study is to develop an automated direction setting algorithm (ADSA) for a smart exit sign system. The challenges of the ADSA are to translate various space types into nodes on a graph, and to provide directions to the shortest safe egress to any evacuees at any point in a building using publicly available exit signs. We have developed a classification for space types and an ADSA that can exclude unsafe paths and calculate the shortest safe path from unknown points in a building to the closest exit points. We do this by reversing and simplifying the start and end nodes of the typical Dijkstra's algorithm by adding a virtual node. The validity of the ADSA was tested through scenario-based simulations of test cases with various conditions and performance comparison with existing algorithms.

1. Introduction

Navigating the indoor spaces of large and complex buildings such as shopping malls is challenging. Building occupants are often disoriented while they explore these types of buildings, and this issue becomes even more problematic during emergency situations such as fires. Kobes et al. examined how people determined evacuation paths during fire-related emergency situations through a series of tests conducted in a hotel building at night [1]. They found that 56.3% of people determined evacuation paths by using exit signs when there was no smoke; however, 81.8% depended on exit signs for an evacuation path when their visibility was impaired due to smoke. These results show that exit signs are of utmost importance when individuals are put in adverse situations that require an exit strategy.

However, exit signs currently installed in buildings have fixed directions, which raise the possibility that these signs could lead evacuees to dangerous areas. For example, a perception test conducted by Choi indicated that the ambiguity in directions on exit signs could create difficulties for evacuees in recognizing the correct direction towards an exit during an emergency situation [2]. Jang et al. also used a simple case study to argue that an evacuation guidance system with a fixed direction image could lead evacuees to dangerous areas [3]. In addition, if the evacuation time is prolonged due to disorientation issues or misleading signs and exceeds the "golden time"—the first few minutes essential for saving a critical mass of lives during evacuation [4]—then the rate of survival critically decreases.

One response to these issues has been mobile-based evacuation guidance systems [5–9]. However, mobile-based evacuation guidance systems are of little use if the evacuation guidance app with a building map has not been installed on the mobile device prior to the occurrence of fire. Another response has been exit-sign-based evacuation guidance systems [10,11], referred to as "smart exit sign" systems or "animated exit signs" [12], where the exit sign can dynamically change its direction and guide evacuees to safe evacuation paths. Existing smart exit sign systems, however, are presently only patents or rough proposals and many gaps still require filling in to make them a practical system. This is because the current systems simply claim that they will use existing shortest-path algorithms and do not include close examinations or validation of the proposed methods.

This study aims to fill in these gaps and add details missing from the shortest-path algorithm required to make a smart exit sign a practical solution. We refer to the algorithm as an automated direction setting algorithm (ADSA) and define ADSA as the algorithm designed to trigger dynamic changes in the direction signs on a smart exit sign to reflect real-time fire situations, and to inform evacuees of the directions towards the shortest safe evacuation paths. The algorithm required for a

* Corresponding author at: Dept. of Architectural Engineering, Yonsei University, A512 Engineering Hall I, 50 Yonsei-ro, Seodaemun-Gu, Seoul 120–749, Republic of Korea. Tel.: +82 82 2 2123 5785.
E-mail address: glee@yonsei.ac.kr (G. Lee).
URL: http://big.yonsei.ac.kr/ (G. Lee).

http://dx.doi.org/10.1016/j.autcon.2015.05.004
0926-5805/

smart exit sign differs from general shortest-path algorithms or a mobile evacuation guidance system in several ways:

- The algorithm should be able to provide the evacuees with directions not only to the closest exit but through the shortest safe egress route at any point in a building while excluding the unsafe areas.
- If the mobile-based evacuation guidance system were a personalized car navigation system, the smart exit sign is a public smart traffic signal that can control traffic flow reflecting the traffic situation. The characteristics of a public system should be considered in developing a smart exit sign system, rather than those of a personalized system.
- The classification of node types is required for proper translation of spaces and elements in a real building into nodes on a graph. For example, not every end node in a network is an exit because some nodes may be dead-ends without exits. Conversely, some spaces without an exit door, but with escape equipment or fireproof materials, may also be considered an exit node.

This paper proposes the requirements of the ADSA for smart exit sign systems by exploring and discussing the above issues, and validating the ADSA through computer simulations. This paper is organized as follows. First, we review commonly used shortest-path algorithms and previous proposals for smart exit sign systems, and discuss their limitations. We then describe ADSA in detail with an illustrated step-by-step example. Finally, we test the applicability and validity of ADSA through scenario-based simulations of a maze case and a real subway station case.

2. Literature review

This section provides a review of the commonly used shortest-path algorithms, followed by existing systems relevant to smart exit sign systems.

2.1. Common shortest-path algorithms

Graph (network) theory [13] describes a path as a continuous sequence of edges connected by vertices. A plain graph calculates the distance between two nodes by finding the smallest number of nodes between them, while ignoring the length of edges. It also does not distinguish between start and end nodes. However, the corridor length is important in evacuation and wayfinding, as is directionality; thus, a weighted directed graph (digraph) is typically used in wayfinding.

The shortest-path algorithm is the algorithm used for calculating the shortest distance between two nodes. This algorithm can be categorized into four types according to the start–end relations in the graphs [14]. Table 1 summarizes these four types of shortest-path algorithms: the single-pair shortest-path (SPSP) algorithm for single start–multiple destination shortest-path problems; the single-source shortest-path (SSSP) algorithm for single start–multiple destination problem; the all-pairs shortest-path (APSP) algorithm for all the pairs in the network; and the single-destination shortest-path (SDSP) algorithm for multiple start–single destination problem. The A* algorithm [15,16] is a representative SPSP algorithm, while Dijkstra's algorithm [17] and the Bellman–Ford algorithm [18,19] are, respectively, representative SSSP and SDSP algorithms, and the Floyd–Warshall algorithm [20,21] is a representative APSP algorithm.

Table 1
The four types of shortest-path algorithms.

Type	Representative algorithms	Description
Single-pair shortest path (SPSP)	A* algorithm	Shortest path finding between a single start point and a single destination point
Single-source shortest paths (SSSP)	Dijkstra's algorithm Bellman–Ford algorithm	Shortest path finding between a single start point and multiple destination points
All-pairs shortest paths (APSP)	Floyd–Warshall algorithm	Shortest path finding between all the possible combinations of single start points and destination points
Single-destination shortest paths (SDSP)	Dijkstra's algorithm Bellman–Ford algorithm	Shortest path finding between multiple start points and a single destination point

The A* algorithm is unsuitable for ADSA because ADSA requires path calculation between multiple start and multiple destination points.

Dijkstra's algorithm and the Bellman–Ford algorithm are similar, except that the Bellman–Ford algorithm allows negative weight values [22]. However, a negative weight value means a negative distance in wayfinding, which does not exist in a real world case. Therefore, the Bellman–Ford algorithm is also unsuitable for ADSA.

Dijkstra's algorithm calculates the shortest path between a single point to multiple points, or *vice versa*. Dijkstra's algorithm, as is, is unsuitable for ADSA because it only generates the shortest paths between nodes in one-to-many relations and does not have a notion of exits and exit signs.

Another candidate algorithm that can be used as a basis for ADSA is the Floyd–Warshall algorithm, which is used when calculating the shortest paths between all possible combinations of a start point and a destination point in a graph. In layman's terms, the Floyd–Warshall algorithm can be understood as repeating Dijkstra's algorithm until it loops through all the nodes in a graph. Therefore, development of ADSA based on the Floyd–Warshall algorithm would be overkill; that is, a much more efficient strategy would be to repeat Dijkstra's algorithm only a limited number of times rather than for all nodes. Considering all these alternatives, our decision was to develop ADSA based on Dijkstra's algorithm rather than using the Floyd–Warshall algorithm.

2.2. Previous studies on evacuation guidance systems

Several evacuation guidance systems have been proposed to provide particular or anonymous evacuees with safe evacuation paths. For example, Lee et al. proposed a smartphone-based evacuation guidance system whereby evacuees could access their own mobile phones for fire and safe egress information detected through a sensor network [9]. The smartphone-based evacuation guidance system has a strength in that it can provide evacuees with visualized fire and safe egress information customized for each user. However, this system will not work if evacuees are not carrying a smartphone or if the evacuation guidance app, which includes a building map, has not been installed on the phone prior to the occurrence of the fire. These indoor navigation systems, including smartphone-based evacuation guidance systems, also often use unique patterns, called “fingerprints,” created by Wi-Fi signals. If Wi-Fi devices are not functioning properly due to automatic electricity shutdown or damage during a fire, a smartphone-based evacuation guidance system will not work even if an app is installed.

For this reason, several conceptual smart exit sign systems have been proposed as alternatives to mobile-based evacuation guidance systems. One patent describes overall functions and a general configuration of a smart exit sign system, without detailed descriptions on how the system determines the directions towards the closest safe exits [23]. Kim et al. proposed what they called an “artificial intelligent directional escape light system [10]” based on the Floyd-Warshall algorithm and another group of researchers proposed a similar system based on Dijkstra's algorithm [11]. However, our previous attempts to develop a smart exit sign system, using Dijkstra's algorithm or the Floyd–Warshall algorithm as they are, revealed several gaps, described in the Introduction, that required filling. The next section describes these gaps and how ADSA operates, in detail, with an illustrated step-by-step example.