An Adaptive Guiding Protocol for Crowd Evacuation Based on Wireless Sensor Networks

Chi-Han Lin†, Po-Yu Chen‡, and Wen-Tsuen Chen†‡
†Department of Computer Science, National Tsing Hua University, Hsinchu 300, Taiwan
‡Institute of Information Science, Academia Sinica, Nankang, Taipei 115, Taiwan
Emails: finalspaceman@gmail.com, poyuchen@iis.sinica.edu.tw, chenwt@iis.sinica.edu.tw

Abstract—Wireless sensor networks have been widely used in many applications. One of them is indoor guiding service for emergency evacuation whose goal is to assist moving objects in escaping a hazardous region safely and quickly when an emergency occurs. In this paper, we propose a distributed and adaptive guiding protocol that takes several factors such as hazardous regions, distance to exits, width of exits, and congestion degree of each location into consideration. This protocol guides moving objects with load balancing among multiple navigation paths to multiple exits and avoids congestion to reduce the evacuation time. Simulation results show that the proposed protocol can guide moving objects (e.g., people) to exits in shorter time and have higher survival rate in comparison to those without balancing the traffic load on navigation paths and exits.

Keywords: wireless sensor networks; emergency guiding; navigation; evacuation; load balance

I. INTRODUCTION

A wireless sensor network (WSN) is composed of many small and autonomous sensor nodes. Each node has the capability of sensing data, processing the collected data, and exchanging data with other nodes. Benefitting from the above features, WSNs have been exploited for various purposes, such as environmental monitoring, vehicular traffic estimation, and healthcare. One of possible applications using WSNs is to provide indoor guiding service, which first constructs proper paths for moving objects (e.g., people) to their destinations and subsequently informs them the guiding directions from their present positions. For example, assume that sensor nodes are deployed in an area of interest in advance. When an emergency occurs in this area, the service can determine the emergency location from the data collected by sensor nodes. It then plans safer paths that detour around the hazardous regions and arranges proper paths that can evacuate all people in short time. According to guiding directions of sensor nodes, a person can follow the planned paths to escape to safe area.

Several previous works proposed centralized guiding protocols that use one or more central servers to monitor the states of all deployed sensor nodes [1]–[4]. Works in [1]–[3] designed schemes to guide mobile robots toward their destinations. Sensor nodes are deployed in an area of interest and periodically transmit the sensing data to central stations, which base on the collected information to plan suitable navigation paths. In [4], a centralized load-balancing guiding system for emergency was proposed. The authors first design an analytical model to estimate the total evacuation time. Then, the base station uses this model to construct a guiding tree whose evacuation time is the shortest. The above centralized guiding protocols rely on wired infrastructure or multi-hop wireless sensor network for gathering sensing data. The central server thus has the comprehensive knowledge of all sensor nodes. Optimal navigation paths are then constructed by the central server and sent via the sensor network to inform and guide the moving objects. However, an emergency (e.g., fire, earthquake) may damage the central server or the sensor network. In other words, centralized guiding systems cannot provide needed reliability in emergency.

Works in [5]–[8] adopted distributed approaches, in which each sensor node exchanges sensing data with neighboring nodes and determines its guiding direction based on collected data. The distributed algorithm proposed in [5] uses the concept of potential to attract or repulse the moving objects. A danger area has a repulsive potential to push away moving objects, and a safe area has an attractive potential to pull them towards it. Based on potential of various locations, navigation paths with guiding directions are constructed for objects to move away from the danger area. The concept of potential is also utilized in [6], which additionally considers scenarios with multiple emergency events and multiple exits in their proposed protocol. Another distributed navigation algorithm is proposed to partition the WSN into several sub-graphs, called skeleton graph [7]. It then uses breadth-first search to find the shortest path on the skeleton graph. This algorithm can transform a dense network into a sparse network and thus can adapt to a scalable sensor network. However, these works may guide moving objects to the same paths or regions, which results in congestion at these places. This situation, also called congestion problem, may prolong the evacuation time. Works in [8] investigated the congestion problem of guiding protocol for evacuation. The number of moving objects at a location is considered as its load. Then, the authors proposed a distributed load balancing guiding protocol that balances the number of people in the entire WSN to alleviate congestion. Nevertheless, it assumes that each exit and path has the same width and capacity. People may still congest easily at a narrow path or near an exit with small width.

In this paper, we propose a distributed and adaptive guiding protocol, which can balance the load among paths and exits in the area of interest and guide people to move around hazardous
regions to exits with various widths. Each location is assigned a potential that is defined as a function of distance to exits, width of exits, and crowd density. This protocol utilizes the concept of potential that a location with a lower crowd density and shorter distance to an exit has a lower potential. The moving objects are evacuated toward the location with lower potential, which implies that they are always guided to less congested paths and finally reach exits. Our proposed guiding protocol has been shown through simulation to evacuate people in the speediest way and rescue the most people from an emergency event, as compared with other protocols.

In Section II, the objective and assumptions of the proposed guiding protocol are described. Then, we define some notations and detail our proposed protocol. In Section III, we compare our protocol with two other distributed guiding protocols in two scenarios. Section IV concludes this paper.

II. THE PROPOSED GUIDING PROTOCOL

We assume a 2D indoor environment where sensor nodes are equidistantly deployed in the corridors, junctions, and exits, as shown in Fig. 1. Thus, the hop count is treated as the distance in this paper. Each sensor node can communicate with other nodes by wireless radio and can display a guiding direction for moving objects to assist their escape from dangers. The proposed protocol dynamically changes guiding directions according to several factors including the distance to exits, the width of exits, the congestion degree of each location, and the location of hazardous regions. This protocol aims to evacuate moving objects as quickly and safely as possible.

A. Assumptions

Suppose that each exit node (i.e., the sensor node located at an exit) knows the width of the exit, which is defined as the maximum number of persons who can pass the exit simultaneously. In addition, each sensor node knows its own walkable area of sensing region and is also assumed to have the IDs of its neighbors. We call a sensor node a neighbor of sensor node \( i \) if it is located within node \( i \)’s communication range and can be reachable in one hop from node \( i \). In addition, each node is assumed to have the capability of sensing the number of people within its sensing range. This could be accomplished by applying image processing [9], [10] or RFID [11] techniques.

B. Definition

In our protocol, each sensor node needs to maintain two tables, namely exit table and neighbor table. The exit table records the information of all exits, whose entry format is \(<\text{Exit ID}, \text{Exit W}, \text{SP Hop Count}, \text{Next Hop}>\). Exit ID and Exit W record the ID and the width of the exit respectively. SP Hop Count and Next Hop are respectively the smallest hop count to the exit and the ID of its next hop node on path with the smallest hop count to the exit. The neighbor table maintains the information of all neighbors with entry format \(<\text{NB ID}, \text{Dest Exit}, \text{Hop Count}, \text{NB Type}, \text{People Num}, \text{Ptl}>\). NB ID is the ID of a neighbor node. DestExit and Hop Count represent ID of the destination exit that the neighbor chooses and the distance in hop count to the destination exit respectively. NB Type indicates type of the neighbor: downstream, upstream, reversal, emergency, or remaining. A downstream neighbor (or downstream node) of sensor node \( i \) represents that node \( i \) selects the neighbor as its immediate evacuation destination. A neighbor whose downstream node is node \( i \) is called upstream neighbor/node of node \( i \). A reversal neighbor is a special type of neighbors for preventing two sensor nodes guide people to each other, which will be discussed in Section III-D. An emergency neighbor of node \( i \) means that the neighbor detects an emergency, and the remaining neighbors are all called remaining neighbors. The field NB Type is initially set to remaining. People Num records the number of people detected by the neighbor, and Ptl is the neighbor’s potential. The default values of People_Num and Ptl are 0.

For convenience, we define the notations used in this paper as follows:

- \( A_i \): The walkable area of sensor node \( i \)’s sensing region.
- \( \text{width}_e \): The width of exit \( e \).
- \( \text{hop}_{i,e} \): The hop count from sensor node \( i \) to exit node \( e \).
- \( N_i \): The number of people that sensor node \( i \) detects within its sensing range.
- \( \text{NB}_i \): The set of all neighbors of sensor node \( i \).
- \( d_i \): The downstream neighbor of sensor node \( i \).
- \( U_i \): The set of upstream neighbors of sensor node \( i \).
- \( O_i \): The set of neighbors of sensor node \( i \) that are neither the downstream node nor upstream nodes (i.e., \( O_i = \text{NB}_i - U_i \)).
- \((i,d_i)\): The guiding direction of sensor node \( i \) from \( i \) to its downstream node. A sequence of guiding directions \((i, d_{i_1}), (i, d_{i_2}), ..., (i, d_{i_n})\), which starts from sensor node \( i \) and ends at exit \( e \) via \( i_1, i_2, ..., i_n \), forms a navigation path for node \( i \) to exit \( e \).
- \( P_i \): The potential of sensor node \( i \).
- \( \text{loc}_P_i \): The location-related potential of sensor node \( i \).

With information stored in sensor node \( i \)’s exit and neighbor tables, \( P_i \) is defined by the following equation:

\[
P_i = \text{loc}_P_i + \alpha \times P_n + \frac{N_i + \beta \sum_{j=1}^{N_i} N_j + \gamma \sum_{l=1}^{N_i} N_l}{A_i},
\]

where

\[
\text{loc}_P_i = -\sum_{e} \frac{\text{width}_e}{(\text{hop}_{i,e} + 1)^2}.
\]
The design of $loc_P_i$ adopts the concept of gravitational potential that is inversely proportional to the square of the distance. A smaller $loc_P_i$ means that the location of sensor node $i$ is closer to an exit $e$ with larger $width_e$. In (1), the potential of $d_i$ (i.e., $P_d$) is added to $P_i$ so that $P_i$ is the potential of entire navigation path from node $i$ to the exit. It is because $P_d$ also contains the potential of $d_i$'s downstream neighbor that recursively reflects potential of every node along the navigation path to the exit. $\alpha$ with value between 0 and 1 is a system weighting factor for $P_d$. The final term in (1) takes into account of node $i$'s congestion degree, which is defined as a function of node $i$'s crowd density within its sensing range and the number of people around node $i$'s neighbors, including node $j \in U_i$ and node $k \in O_i$. Node $j \in U_i$ and node $k \in O_i$ are taken into consideration since the upstream neighbors are guiding people toward node $i$ and neighbors in $O_i$ may change their downstream neighbors to node $i$ at a later time. When node $i$'s congestion degree increases, node $i$ raises $P_i$ in order to avoid overcrowded people being guided toward it. $\beta$ and $\gamma$ with values between 0 and 1 are weighting factors for the number of neighboring people. In summary, if 1) sensor node $i$ is closer to an exit with larger width, 2) its crowd density is smaller, 3) the number of people of its neighbors is fewer, and 4) the potential of its downstream node is lower, $P_i$ is smaller.

Finally, we define three types of packets used for communication among sensor nodes. The first is $initial$ packet for delivering information of exits with the format <$Exit_ID$, $Exit_W$, $Sender_ID$, $Hop_Count$>. $Sender_ID$ denotes sender's ID of the packet, and $Hop_Count$ represents its current hop count to the exit. The second is $update$ packet that contains the information of a sensor node with the format <$Sender_ID$, $DStream_ID$, $Dest Exit$, $People_Num$, $Ptl$, $Exit_Table$>. $DStream_ID$ is the ID of the downstream node that the sender chooses. $Dest Exit$ and $People_Num$ respectively represent ID of the destination exit that the sender chooses and the number of people detected by the sender. $Ptl$ and $Exit_Table$ record the sender's potential and exit table respectively. The third is $emg$ packet with the format <$Sender_ID$>, which is transmitted when a sensor node detects an emergency.

**C. Protocol Details**

The proposed guiding protocol consists of two phases: initialization phase and potential management phase. When deployment of the guiding system finishes, the initialization phase starts. Thus, the initialization phase is finished before an emergency occurs. In this phase, all sensor nodes through packet exchanging know the information of all exits and determine their initial downstream neighbors. In potential management phase, each sensor node first calculates its potential according to (1). Then, it exchanges its potential and other information with its neighbors and re-selects a neighbor with the smallest potential to be its downstream neighbor. When a sensor node detects an emergency, it immediately informs its neighbors to avoid moving objects being guided toward it.

**Initialization Phase:**
1) An exit node $e$ in the environment broadcasts an $initial(e, width_e, 0)$ packet.
2) If sensor node $i$ receives exit $e$'s $initial(e, width_e, j, hop_{e+1})$ packet the first time from its neighbor node $j$, go to step 3; otherwise, go to step 4.
3) Node $i$ inserts a new entry of <$e, width_e, j, hop_{e+1}$> in its exit table. Go to step 5.
4) If exit $e$'s SP_Hop_Count stored in node $i$'s exit table is larger than $hop_{e+1}$, node $i$ updates the entry to <$e, width_e, j, hop_{e+1}$> in the exit table.
5) Node $i$ sends an $initial(e, width_e, i, hop_e = hop_{e+1})$ packet to its neighbors. Return to step 2.

After each sensor node collects the information of all exits, it chooses a closest exit as its destination exit and sets the neighbor with smallest hop count to the exit to its downstream neighbor in its neighbor table. Then, potential management phase is activated. However, in a distributed system, each sensor node is difficult to determine whether the information of all exits is completely collected because network delay is unexpectable. Here, we use a time threshold to solve this problem. If a sensor node does not receive an initial packet that could update its exit table within the time threshold, the node completes the collection of exit information and executes subsequent procedure mentioned above.

**Potential Management Phase:**

This phase is described in terms of sending and receiving a packet. The sending procedure of a node is as follows:
1) Node $i$ computes its potential $P_i$.
2) If the downstream neighbor, exit table, or $P_i$ changes, node $i$ sends an $update(i, d_i, Dest Exit_i, N_i, P_i, Exit_Table_i)$ packet to its neighbors.
3) If node $i$ detects an emergency within its sensing range, node $i$ sends an $emg(i)$ packet to its neighbors.
4) Return to step 1.

The procedure of a node receiving an $update$ packet is as follows:
1) Node $i$ listens wireless signal to wait to receive an $update(j, d_j, Dest Exit_j, N_j, P_j, Exit_Table_j)$ packet from its neighbor node $j$. If node $i$ receives it, go to step 2.
2) Node $i$ updates the corresponding fields in its neighbor table and stores Exit_Table.
3) Node $i$ checks whether Exit_Table_i; lengthens its original smallest hop count to an exit or not. If the above is true, node $i$ updates its exit table.
4) Node $i$ changes node $j$'s NB_Type in node $i$'s neighbor table according to the original NB_Type of node $j$ and whether node $i$ is definitely $d_i$ or not, as shown in Fig. 2.
5) Node $i$ re-determines its downstream neighbor by the downstream node decision scheme detailed in Section III-D. Return to step 1.

The procedure of a node receiving an $emg$ packet is described as follows:
1) Node $i$ listens wireless signal to wait to receive an $emg(j)$ packet from its neighbor node $j$. If node $i$ receives it, go to step 2.
2) Node $i$ changes node $j$'s NB_Type to emergency to prevent choosing node $j$ as its downstream neighbor.
3) If the emergency neighbor $j$ lengthens its original shortest hop count to an exit, node $i$ updates its exit table.
Node $i$ re-determines its downstream neighbor by the downstream node decision scheme detailed in Section III-D. Return to step 1.

### D. Downstream Node Decision Scheme

The following steps describe how to determine the downstream node of a sensor node:

1. Node $i$ chooses a neighbor with lowest potential from its downstream and remaining neighbors in node $i$’s neighbor table as its downstream neighbor. If node $i$ has no such neighbor, go to step 2.
2. Node $i$ chooses a neighbor with lowest potential from its upstream neighbors as its downstream neighbor. If node $i$ has no such neighbor, go to step 3.
3. Node $i$ chooses a neighbor with lowest potential from its reversal neighbors as its downstream neighbor. If node $i$ has no such neighbor, node $i$ is surrounded by emergency.

In general case, each sensor node always chooses the downstream neighbor from its downstream and remaining neighbors. Unfortunately, some sensor nodes only have upstream neighbors since an emergency occurs so that these nodes lose their downstream neighbors. In step 2, a node $i$ chooses an upstream neighbor $j$ as its downstream neighbor and then sends an update($i, d_i = j, \text{Dest Exit}, N_i, P_i, \text{Exit Table}$) packet to node $j$. Node $j$ receives the update packet from its downstream neighbor $i$ and then sets node $i$’s NB Type to reversal according to Fig. 2. A reversal neighbor of node $j$ means that node $j$ has an upstream neighbor that is the original downstream neighbor of node $j$.

### E. Message Overhead Reduction

In potential management phase, a sensor node transmits an update packet to its neighbors once the downstream neighbor, exit table, or potential changes. However, a sensor node does not frequently change its downstream neighbor since the potential does not vary significantly in general case. Therefore, the number of update packets generated from changing downstream node is limited.

As mentioned in Section III-C, a sensor node updates its exit table only when the emergency lengthens its shortest path to exits. Furthermore, the exit table of each sensor node is unchanging when there is no emergency in the environment.

Hence, the number of update packets generated from modifying exit table is also limited. Nevertheless, the potential changes very frequently since the people are always moving in the environment. If an update packet is sent once the potential varies, the message overhead will be enormous. For the sake of reducing the message overhead, a threshold $T_p$ is defined to restrict sending update packets. If the potential of node $i$ (i.e., $P_i$) changes, node $i$ checks the difference between $P_i$ and the potential that node $i$ sent last time (denoted as $P'_i$). If $|P_i - P'_i| > T_p$, node $i$ transmits an update packet to its neighbors.

### III. PERFORMANCE EVALUATION

In this section, the performance of the adaptive guiding protocol (AGP) is evaluated by simulation and is compared with two other distributed guiding protocols. One is the shortest path based protocol (SP) [5], [6], and the other is load-balanced navigation protocol (LBNP) [8].

#### A. Simulation Parameters and Scenarios

In our simulation, 10 × 10 sensor nodes are deployed equidistantly in 100m × 100m indoor environment, as shown in Fig. 3. Three exits with different widths are located in upper left, upper right, and lower right, respectively. Their widths are respectively 2m, 4m, and 6m. The total width of all roads is set to 30m, and the width of each road is between 2m and 5m. The maximum moving velocity of each person (i.e., the moving object in the simulation) is 5m/s, and the data transmission rate of each sensor node is 250Kbps. The packet loss rate is 5%, and $\alpha$, $\beta$, $\gamma$, and $T_p$ are set to 0.5, 0.5, 0.2 and 0.2, respectively.

In addition, our simulation has two scenarios. The first scenario is in a non-emergency environment. The guiding time and the standard deviation of the load of each exit are evaluated in this scenario. The guiding time is defined as the demand time from the start of evacuation to the last person escaping from the environment. The load of an exit is defined as the number of people escaping from it divided by its width. The second scenario is in a diffusible emergency environment (e.g., a fire). An emergency occurs at a random node and then starts to diffuse to one of its neighbors every 5s. Suppressing the emergency is not considered. Moreover, each person is assumed to be able to survive within the emergency region up to 25s. In this scenario, survival rate becomes the main metric, defined as the ratio of people escaping from the environment successfully. In each scenario, each case is run 100 times.
B. Simulation Results

In non-emergency environment, the guiding time of AGP, as compared with two other protocols, is the shortest in all cases, which is demonstrated in Fig. 4. When the number of people is few, the guiding time of three protocols is very close. As the number of people increases, the guiding time gradually increases, especially in using SP. The reason is that SP does not take the congestion into consideration, which makes congestion occur easily in some areas. Although the LBNP can balance the load of each node, it cannot balance the load of each exit and may easily cause congestion near an exit with small width. As a result, AGP outperforms LBNP about 30 seconds in most cases, and it saves about 50-130s while comparing with SP.

Fig. 5 shows the standard deviation of the load of each exit, denoted as STA(E). AGP has the smallest STA(E) and keeps STA(E) about 20-25 in all cases. It implies that AGP significantly balances the load of each exit. This phenomenon also can explain that AGP defeats two other protocols at the metric of guiding time. The STA(E) of LBNP is higher than AGP because LBNP always tends to guide the same number of people to each exit. A large STA(E) indicates that too many people leave from some exits with small width but too few people leave from large width exits. As a consequence, people congest near an exit with small width and prolong guiding time.

In diffusible emergency scenario, the survival rate decreases gradually while the number of people increases, as observed from Fig. 6, since evacuating more people needs to spend more time and the emergency area is enlarged over time. However, AGP still achieves the highest survival rate. It keeps at least 98.5% even when the environment is very dense.

![Fig. 4. Comparison of guiding time](image)

![Fig. 5. Standard deviation of the load of each exit](image)

IV. CONCLUSIONS

An adaptive and distributed guiding protocol is proposed in this paper, which considers the load balancing of each navigation path and exit. Traditional centralized guiding protocols cannot provide the reliability for emergency, and previous distributed protocols guide people on the shortest path to exits but neglect the congestion problem. Our proposed protocol can avoid people congestion to shorten the guiding time. From simulation results, this guiding protocol achieves the shortest guiding time and the highest survival rate in non-emergency environment and emergency environment. This protocol outperforms the conventional shortest path based protocols and the load balanced navigation protocols due to considering more realistic factors in practical environment.

REFERENCES