Emergency Evacuation using Wireless Sensor Networks

Matthew Barnes*, Hugh Leather*, D. K. Arvind
Research Consortium in Speckled Computing
School of Informatics, University of Edinburgh
Email: { mbarnes, hleather, dka } @inf.ed.ac.uk
*Corresponding Authors

***Abstract*—This paper presents a distributed algorithm to direct evacuees to exits through arbitrarily complex building layouts in emergency situations. The algorithm finds the safest paths for evacuees taking into account predictions of the relative movements of hazards, such as fires, and evacuees. The algorithm is demonstrated on a 64 node wireless sensor network test platform and in simulation. The results of simulations are shown to demonstrate the navigation paths found by the algorithm.**

I. Introduction

Emergency navigation through indoor environments is a novel application area for the deployment of wireless sensor networks (WSN). In this paper we describe a distributed algorithm for guidance of escaping occupants in the event of a hazard in a building. In [1], [2] and [3] navigation techniques are proposed for wireless sensor networks for obstacle and hazard avoidance. [2] and [3] focus on emergency applications using a navigation algorithm to avoid hazardous regions and direct escaping persons towards safe locations.

The FireGrid project [4] aims to use WSN in buildings in conjunction with Grid Computing to monitor buildings. The WSN provides input to super-real-time simulation of the spread of fire and smoke throughout the building to aid the emergency response. To use such simulations for assisting navigation from each sensor location, all data must be retrieved from the network, collated and input to the simulation before returning the navigation information back to individual nodes. That approach may be costly in time and require a large amount of network activity to disseminate the navigation information throughout the building.

Offline modelling and simulation of fire spread and occupant evacuation is a research area in fire safety and architectural design. Different modelling techniques can be applied to each and are reviewed in [5].

This paper presents a new evacuation assistance application for wireless sensor networks that differs from previously proposed algorithms. By using a predetermined model of the building, distributed throughout the network, safe escape routes are found. These routes maximise the time an evacuee with remain ahead of the hazard while escaping along the path. By using a simple model we aim to provide immediate and safe navigation assistance in the event of a hazard through distributed route finding.

In Section II we review existing work in the field. In Section III we describe the algorithm and model in detail. Results from simulation and a prototype implementation are shown in Section IV. Further work are discussed in Section V before we conclude in Section VI

II. Related Work

This section briefly reviews existing work on emergency guidance and navigation algorithms using wireless sensor networks. The works of Li et al.[1], Tseng et al.[2] and Corke et al.[6] use wireless sensor networks to navigate a 2D plane. In [6] and [1] the concept of artificial potential fields is used for flying robot navigation over a network and emergency navigation.

In [1] an approach is proposed to find safe paths through a network that can contain multiple sources of danger (known as obstacles) to an exit area. The exit area generates a positive potential to attract the navigating user, whilst obstacles generate a repulsive potential so that safe paths will pass around them. A limitation of the algorithm proposed in [1] is the lack of concept of the size of a hazardous area, this is addressed by [2] by applying a 'hazardous region' around the source of a hazard and routing around these regions.

In [2] the shortest exit path for each node is computed in an initialisation phase; the area affected by a detected hazard is fixed, paths are adjusted to avoid such areas. In the event of an emergency, each node already knows where to direct evacuees and only nodes whose path passes through the hazardous region are affected. This reduces the transmission overhead as only nodes that are affected by the hazardous region need be re-evaluated. As a result the network settles and safe paths are found from each location faster.

In [3] an extension to the work previously shown in [2] is proposed for an emergency guidance service for 3-dimensional building environments through the addition of special stairway nodes to connect isolated floors. The stairway nodes effectively act as exit nodes linking between floors. The drawback of this approach is that each floor is isolated and hazardous zones only extend between floors through these stairway nodes. This does not takes into account that a hazard on a floor above or below may affect a location more rapidly than through a path via a stairwell. For example a fire may affect areas on floors above and below that on which it first takes affect.

The algorithm proposed in our paper uses a model of the

hazard's spread that can be specified on arbitrary graphs, thus handling three dimensional cases in a natural manner, without the addition of specialised nodes or concepts such as isolated floors. The entire network can be linked through any available path for a hazard to spread. The presence of hazards throughout is taken into account in finding exit paths from all locations.

Our algorithm models the progress of the hazard and the progress of the evacuees to ensure the evacuees stay safely ahead of the hazard. This is in as opposed to considering only static regions over which no concept of time is defined. Section III will discuss our algorithm in detail.

III. Emergency Navigation Application

In this section we describe our approach to the problem of safe navigation. We present the distributed navigation algorithm and the model of the building environment that is used.

A. Building Model

The proposed algorithm uses a predetermined model of the spread of hazardous conditions and the speed of evacuation of persons from the building. [5] has a survey of offline modelling and simulation techniques for fire safety analysis and engineering. The information required by our algorithm is comparable to a 'Detector Response Model'[5] of a fire spreading. The output from such simulation could be used to provide the estimated timings for our model to work upon. In this paper we make no distinction as to the nature of the hazard only the rate at which the harmful effects may spread.

The building environment is specified as two weighted graphs over the network, called the 'hazard graph' and the 'navigation graph'. The nodes of each graph are sensor locations, edges between describing the movement of hazards and the movement of people.

The nodes in the hazard graph are connected by an edge if the hazard may spread directly from one node to another. The weight of the edge is the fastest time taken for a hazard to traverse the edge. We expect these weights to be measured from accurate offline hazard simulations, or from the best conservative estimates of emergency engineers. Edges may be directed to take account that the hazard may spread at different rates in different directions; for example, a fire may spread upward through a building at a greater rate than downwards.

Fig 1 shows an example of a hazard graph projected over a section of a building floorplan. It demonstrates that hazards may spread through walls and along corridors. Once any particular node is in hazard, the neighbours of that node will follow by as many time units later as are shown on the edges between them. The navigation graph is formed of the human navigable paths between sensor locations. Edges between nodes are weighted with the time taken for an evacuating person to travel between the locations by the navigable path. The navigation graph may also be directed as the time taken to move between two locations may depend on the direction travelled since, for example, fleeing upstairs takes longer than fleeing downstairs.

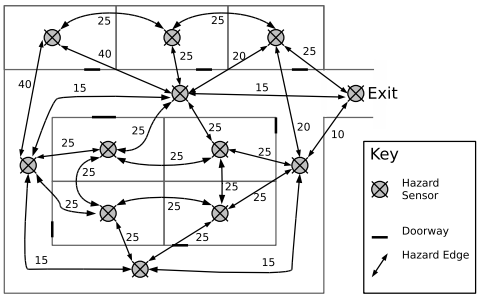


Fig. 1. Example Hazard Graph

Fig. 2 shows an example of a navigation graph. One may see in this particular graph that nodes are only connected if there is a door or open corridor between them.

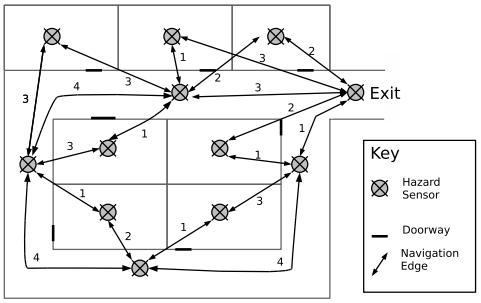


Fig. 2. Example Navigation Graph

When referring to the hazard and navigation graphs, we use the following notations:

- Hazard weight, $F_{u,v}$: this is the time taken for a hazard to spread from the location of sensor u to that of sensor v.
- Navigation weight, $R_{u,v}$: this is the time taken for a human to navigate from the location of sensor u to that of sensor v.
- V is the set of all sensor nodes throughout the network.

Estimates used are conservative, with the worst cases being used. The weighted edges of the hazard graph are the fastest possible times for the hazard to spread via that edge. The edges of the navigation graph are the slowest time for a person to travel between the locations.

The hazard and navigation graphs may be very different as many more routes may exist by which hazard may spread that a human may not travel. This is particularly apparent in buildings with multiple floors where hazards may spread whereas a person may not.

Although we have designed the application to be a fire safety system, the algorithm can be applied to other hazardous conditions by straightforward means.

B. Hazard Time Computation

Before computing the safe paths out of the building, we need to predict the spread of the hazard. Each node must maintain

and compute the time at which its location will be hazardous. We call this the 'hazard time' and introduce the following notation:

- Hazard time H_v: this is the estimated time until the location of sensor v will be hazardous.

Formally, H_v is given by:

Let, P be the set of all paths beginning at a node in hazard and ending at v, then

$$H_v = Min_{p \in P, p=\langle p_1, p_2, ..., p_n=v \rangle} (\sum_{i=1}^{n-1} F_{p_i, p_{i+1}}).$$

The hazard times may be trivially computed by breadth first search.

Fig. 3 shows the hazard times computed for the example hazard graph from Fig. 1. The fastest paths for the hazard to spread have been found and each sensor has computed its hazard time.

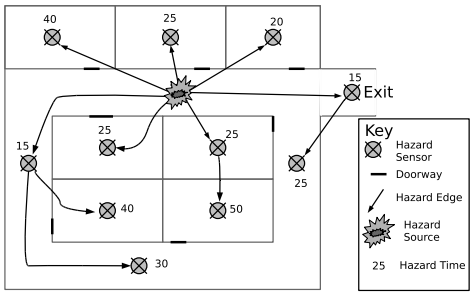


Fig. 3. Computed Hazard Times

C. Escape Path Safety

The safety measure of escape paths is crucially different amongst different works. In [1] the shortest path to an exit that avoids the location at the source of an obstacle is defined as the safest path. In [3] a safe path is one that will reach an exit or safe location without directing an evacuee through a hazardous region defined by a fixed number of sensor locations.

Our navigation algorithm assigns a metric of safety to paths. The safety of a path to an exit node is the minimum of the differences in time between the hazard reaching each node on the path and an evacuee starting at the beginning of the path reaching those nodes. Our metric of safety is therefore the margin of time that an evacuee has to reach each node before the node becomes hazardous.

Formally, the safety of a path is inductively computable. The safety of the empty path at the exit node is the hazard time for the exit node, since an evacuee starting at the exit has as much time as it takes for the hazard to arrive before they have to leave:

$$S(\langle v_{exit} \rangle) = F_{v_{exit}}.$$

Given the safety for some path,

$$p = \langle v_i, v_{i+1}, ..., v_{exit} \rangle.$$

We may prepend a node, v_{i-1}, giving the path

$$p' = \langle v_{i-1}, v_i, v_{i+1}, ..., v_{exit} \rangle.$$

The safety of this extended path is:

$$S(p') = min(S(p) - R_{v_{i-1}, v_i}, H_{v_{i-1}}).$$

The intuition behind the first term of the minimum is that any delay beginning on an evacuation path makes the path less safe by the amount of the delay and that a human is delayed walking to a path, before starting on it.

The second term represents the insight that no matter how safe an adjacent path is, the hazard might still reach an evacuee first.

For any node the safest path is the path to exit with the highest safety (there may several with the same safety value). As such it is possible to speak of the safety of a node being the safety of the safest path from that node. We introduce the notation:

Let, P be the set of all paths beginning at a v and ending at the exit, then

$$S_v = Max_{p \in P} S(p).$$

If the safety of a node is negative, then an evacuee can travel no path from it to the exit without crossing a node in hazard. The more negative the safety of the node, the longer that a node on the exit path will have been of hazard before an evacuee exiting via the path passes through the node in hazard.

Due to the inductive construction of the paths, each node only needs to know its successor to completely determine the safest paths for all nodes in the graph.

An example of the paths and the safeties found by the algorithm for the example graphs shown in Section III-A is detailed below. The hazard times have been computed as shown in Fig. 3. The safest paths to the exit node are shown in Fig. 4.

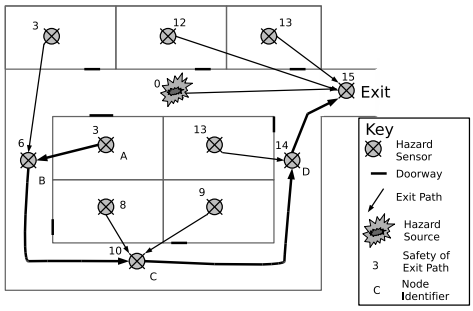


Fig. 4. Examples of Safety of Exit Paths

We examine the highlighted path $\langle v_A, v_B, v_C, v_D, v_{Exit} \rangle$ from node v_A to the exit node. The safety of the exit is calculated first after the hazard time is computed. The safety of the paths from each node on the path to the exit are then found in the reverse order of the exit path from node v_A as shown below.

$S(\langle v_{Exit} \rangle) = H_{v_{Exit}} = 15,$
$S(\langle v_D, v_{Exit} \rangle) =$
$min(H_{v_D}, S(\langle v_{Exit} \rangle) - R_{v_D, v_{Exit}}) =$

$min(25, 15-1) = 14,$
$S(\langle v_C, v_D, v_{Exit}\rangle) =$
$min(H_{v_C}, S(\langle v_D, v_{Exit}\rangle) - R_{v_C,v_D}) =$
$min(30, 14-4) = 10,$
$S(\langle v_B, v_C, v_D, v_{Exit}\rangle) =$
$min(H_{v_B}, S(\langle v_C, v_D, v_{Exit}\rangle) - R_{v_B,v_C}) =$
$min(15, 10-4) = 6,$
$S(\langle v_A, v_B, v_C, v_D, v_{Exit}\rangle) =$
$min(H_{v_A}, S(\langle v_B, v_C, v_D, v_{Exit}\rangle) - R_{v_A,v_B}) =$
$min(25, 6-3) = 3.$

The evacuee has no more than three time units spare on this path. Indeed, it is the exit node that an evacuee is estimated to reach three time units before the hazard. (From Fig. 2 the weight of the path $A, B, C, D, Exit$ is 12 time units, the hazard time of the exit node is 15.)

D. Concurrent Distributed Algorithm

The inductive formulae for hazard time and safety suggest a simple implementation with two phases. One needs first to propagate all hazard times forward from the nodes in hazard to the whole network so that all nodes have correctly found their hazard time. That being complete, the safeties can then be propagated backwards from the exits to each node.

Such a simple implementation is not, however, applicable to wireless sensor networks in general and is prone to failure under emergency conditions. In general it is not possible to bound the time at which each node will receive its hazard time. Network delays are variable at the best of times and cannot be guaranteed in the presence of interference. Moreover, in an emergency, any one node may be destroyed by the hazard! No algorithm, then, that waits for each node to receive the correct hazard time will suffice.

Our algorithm has been developed such that both stages run concurrently following the detection of a hazard. Hazard and safety times are updated continuously as more recent and accurate information is received.

Each node stores a table containing the identifiers of all neighbouring nodes toward which it may direct people and stores the safety of the exit path from each of these nodes.

1) Initialisation: To begin, each node sets its estimates of hazard time, safety and successor node, so that,

$$h_v := \infty,$$
$$s_v := -\infty,$$
$$successor_v := \perp.$$

These estimates will approach the exact values of H_v, S_v and $Successor_v$, respectively, as the algorithm proceeds to a fixed point.

Each node also knows its unique identifier, ID_v.

Additionally, each node sets up a table for the safeties it has received from its neighbours, T_v^u (for each neighbour, u; initially these are all $-\infty$).

2) Emergency Detection: When a sensor node, v, detects an emergency it sets its own hazard time, h_v, to zero and broadcasts a packet containing the data $\{ID_v, h_v, s_v\}$. The node does not yet have any knowledge of paths to exit locations, so its safety remains unchanged. This transmission triggers the network to locally compute how the hazard is expected to spread and find appropriate safe exit paths.

3) Algorithm Operation: The algorithm is reactive to radio transmissions received from neighbouring nodes. The rules governing the operation, on receiving the packet $\{ID_u, h_u, s_u\}$, are shown below for sensor node v:

1: $T_v^u := s_u$
2: **if** $h_u + F_{u,v} < h_v$ **then**
3: $\quad h_v := h_u + F_{u,v}$
4: **end if**
5: **if** $v = v_{Exit}$ **then**
6: $\quad s_v := h_v$
7: **else**
8: $\quad s_v := min(h_v, Max_{w \in V}(T_v^w - R_{v,w}))$
9: $\quad successor_v$ is set to be any neighbouring node w for which $T_v^w - R_{v,w} \geq (T_v^x - R_{v,x})\ \forall\ x \in V$
10: **end if**
11: **if** s_v or h_v has been changed **then**
12: $\quad$ transmit $\{ID_v, h_v, s_v\}$ to all neighbours
13: **end if**

The operation of lines 2-4 is to propagate hazard times throughout the network whilst lines 3-10 propagate and calculate path safeties. To do the latter correctly, we must remember previously received neighbour safeties. This is because as new information comes in the safeties may go up, as safer routes to the exit are found, or down as faster routes for the hazard come in. We must know which neighbour is the second best if the previously best successor is no longer the first node on the safest way out.

E. Network Deployment

Hazard sensors are to be deployed throughout a building to maximise coverage of hazard detection. Additional nodes may be required to form a reliable network for communication. These nodes do not form part of the hazard or navigation graphs and would exist only to route transmission between nodes connected in the graphs where a single hop communication link does not exist. We refer to these as relay nodes.

The sensing capability of each hazard sensor is not explored in this paper as the algorithm can be used to describe the spread of any hazard for which a suitable graph weights can be generated.

Another set of nodes is also required to guide evacuees via the exit paths. These devices are required to interpret the safe routes found through the model; they may take the form of lighted signs, floor lights or individual earpieces for the visually impaired. The operation of these devices is not discussed in this paper.

To facilitate the operation of the algorithm, the data from the hazard and navigation graphs must be distributed throughout the network. At deployment each sensor is assumed to be initialised with the following information:

- A unique global identifier.

- The identifiers of all hazard sensors to which it is connected in the hazard or navigation Graphs.
- The weights of all incoming edges on the hazard graph.
- The weights of all outgoing edges on the navigation graph.
- Whether the sensor is an exit node.

Sensors monitor the environment for hazards and when idle the network can be used for other applications whilst monitoring for hazards. When a hazard is detected other non-critical applications should cease so as to free the network for the navigation algorithm.

IV. Experimental Setup

In this section we describe current work on the simulation of the application and an implementation on wireless sensor network hardware.

A. Physical Implementation

We have implemented the algorithm on the Prospeckz-IIk platform [7]. The Prospeckz-IIk device is a 32mm x 22mm wireless sensor node with Cypress CY8C29666 Programmable System-on-Chip (PSoC) Microcontroller [8] with 32Kbytes of Flash memory and 2Kbytes of Random-Access Memory, a Chipcon CC2420 2.4GHz transceiver [9] and an on-board chip antenna. A small software framework has been developed to facilitate application development, including an implementation of a medium access control protocol, SpeckMAC [10].

Testing of the implementation on a 2 dimensional arrangement of nodes used the Perspeckz64 [11] resource shown in Fig. 5. The Perspeckz64 is a power and programming platform for up to 64 Prospeckz devices arranged in a 8x8 2-dimensional grid with 10 cm separation between adjacent devices. It is web accessible and wirelessly programmable to facilitate the testing of sensor network applications and protocols. The implementation has shown that the application is feasible for resource-limited wireless sensor devices and has been used to verify the results of simulations of 8 x 8 grids of nodes.

B. Simulation Framework

A simulation framework has been developed using the Specksim simulator[12]. SpeckSim is an event-driven simulator for wireless sensor networks written in JAVA.

The Prospeckz devices were modelled in Specksim. The SpeckMAC network stack was implemented upon this base and has been used for all simulations. The simulator is shown in Fig. 6

C. Simulation Results

Table I shows simulation results for a 2-dimensional grid layout of 100 sensors with a variety of exit locations and hazard sources. These simulations used hazard and navigation graphs such that every node is connected by a bidirectional edge to its vertical and horizontal neighbours in the grid layout. The weight of every edge in the hazard graph is set to 5 time units, all edges in the navigation graph are weighted with 1 time unit. Therefore a person travels five times faster than a hazard. For simulations where more than one hazard is detected, the hazards are found at the same instant to demonstrate the applications ability to handle multiple simultaneous hazards.

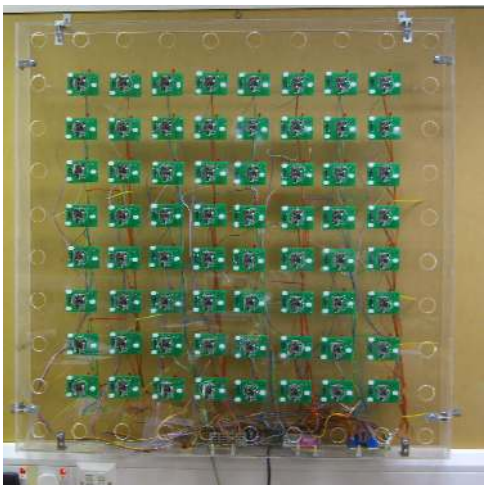


Fig. 5. Perspeckz64 Platform

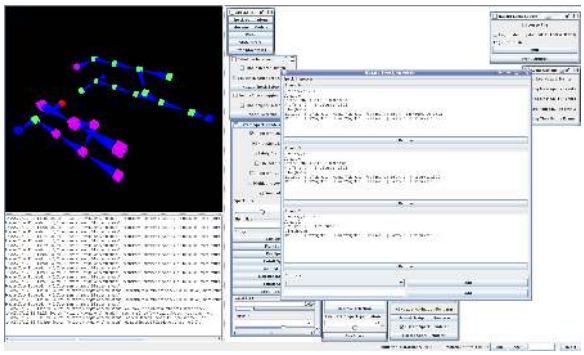


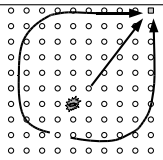
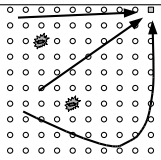
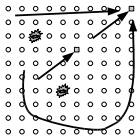
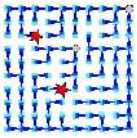
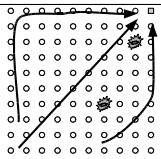
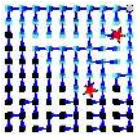
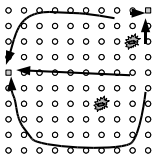
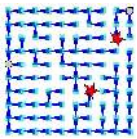
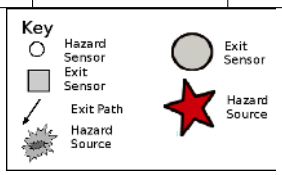
Fig. 6. Simulation Environment

The table shows a simple and clear representation of the directions of the safest exit paths from different areas on the grid and the more detailed but intricate paths found by the simulated network. The path from every node is shown in the simulation results as each node is directing towards its safest neighbour. The number of transmissions required to settle the network and the time taken for the procedure to complete is also given.

The simulation results show that the application does not suffer greatly from additional overhead of transmissions or settling time if multiple hazards are detected. The paths found by the algorithm can be seen to route evacuees around hazardous areas to reach exit nodes if possible.

Example D simulates a hazard detected close to the only exit location. The darker coloured nodes of the simulation results are locations from which it is impossible to reach the exit before the hazard has spread to a node on the path. In this instance the algorithm directs people via a path that minimises their exposure to the hazard by aiming for locations that will

TABLE I
SIMULATION RESULTS

Example	Safest Paths	Simulation Results	Tx	Time (s)	Example	Safest Paths	Simulation Results	Tx	Time (s)
A			461	5.4	B			401	5.7
C			451	4.9	D			421	3.6
E			481	4.4					

have been hazardous for the least possible time.

A second exit is added to the configuration of D in Example E. This exit is much further from the hazard sources so attracts many more paths across the network as people can now exit safely from more locations. Some nodes still direct towards the original exit as they must route around the closer hazard to reach the new exit's location.

D. Implementation Issues

In the implementation of the application on Prospeckz devices it became apparent that the current CSMA MAC protocol is not ideal for this situation. As the application is reliant on radio reception to calculate the correct minimum hazard times and optimum exit paths. A single missed transmission can affect the outcome of a large portion of the network so a requirement for the application is a reliable transport layer.

To counter this problem our implementation transmits periodic update packets to neighbouring nodes so that if a packet is missed it is likely to be received in a subsequent update. This adds to the number of transmissions required to successfully find the exit paths for each node but with a suitable choice of update period the adverse effect on the settling time of the network can be limited. The settling time of the 64 node physical platform remains similar to that of the simulated network and returns the same exit paths.

The network problems experienced in the implementation will be fixed in future applications with the introduction of a reliable and efficient transmission scheme. Due to the somewhat sequential operation of the application and the desire for low latency when finding paths, a scheduled or slotted MAC protocol may be more suitable.

V. FURTHER WORK

There are several areas in which we look to extend the application and address issues that may be crucial for an robust and practical deployment:

- *Communication network and role of relay nodes.* The need for reliable and efficient communication has already been identified; the introduction of relay nodes as discussed before may be necessary, particularly in routing communications between floors. The operation of relay nodes has not been tested to date in our framework. Neither does SpeckMAC provide the routing to make use of them.
- *Routing*. Intelligent routing of hazard time and safety information is necessary to avoid communication bottlenecks on vast graphs. By routing the hazard time to the exit nodes as quickly as possible, paths may be computed early. Routing region approximations could be used to speed up initial estimates of hazard times for exit nodes.
- *Adaptation to the actual rate of spread of a hazard*. If a hazard spreads faster than expected there is no adverse effect on the application as this appears no differently as when multiple hazards are detected. However, we currently do not take any advantage if a hazard spreads at a lower rate than expected. In such a situation there may be safer paths that the algorithm is unable to find.
- *Initialisation*. An initialisation phase can be used to compute the shortest path to an exit from each node whilst the application is idle. On detecting a hazard nodes would immediately have an estimate for the successor on their exit path.
- *Distinction between different forms of danger presented by a hazard*. In the case of a fire there may be two forms of danger to building occupants, flames and smoke, that may spread via different paths and at different rates with complex correlations. The application could be extended to take account of different levels of hazards at locations and route evacuees accordingly.
- *Overlaying multiple human navigation models*. These may take account of the different movement capabilities of people with disabilities.

- Monitoring of the evacuation of building occupants for congestion of routes and adaption of the safety of paths to these situations.
- Validate survival rates as compared to other techniques using detailed offline fire and egress simulation.

VI. Conclusion

We have presented a novel approach to safely evacuating persons from buildings in hazardous conditions. We have shown the implementation in simulation and on physical devices and demonstrated the efficacy of the algorithm. Our approach differs from previous works in its ability to handle arbitrary graphs and by modelling hazard and evacuee speeds through the dangerous environment. The exit paths found by our algorithm are not based solely on static data but adapt to the dynamics of the scenario.

Acknowledgement

This work was supported in part by the Scottish Funding Council under the Strategic Research Development Grant R37329, and UK Engineering and Physical Research Council under the Basic Technology Research Programme, Grant C523881 entitled "Research Consortium in Speckled Consortium"

References

[1] Qun Li, Michael De Rosa, and Daniela Rus, "Distributed algorithm for guiding navigation across a sensor network," in *Proceedings of the Ninth Annual International Conference on Mobile Computing and Networking*. ACM Press, Sep. 2003, pp. 313–325.

[2] Yu-Chee Tseng, Meng-Shiuan Pan, and Yuen-Yung Tsai, "Wireless sensor networks for emergency navigation," *IEEE Computer*, vol. 39, no. 7, pp. 55–62, Jul. 2006.

[3] Meng-Shiuan Pan, Yuen-Yung Tsai, and Yu-Chee Tseng, "Emergency guiding and monitoring applications in indoor 3d environments by wireless sensor networks," *International Journal of Sensor Networks*, vol. 1, no. 1/2, pp. 2–10, 2006.

[4] D Berry, A Usmani, J Terero, A Tate, S McLaughlin, S Potter, A Trew, R Baxter, M Bull, and M Atkinson, "Firegrid: Integrated emergency response and fire safety engineering for the future built environment," UK e-Science Programme All Hands Meeting, Sep. 2005.

[5] Stephen M Olenick and Douglas J Carpenter, "An updated international survey of computer models for fire and smoke," *SFPE Journal of Fire Protection Engineering*, vol. 13, no. 2, pp. 87–110, 2003.

[6] Peter Corke, Ron Peterson, and Daniella Rus, "Networked robots: Flying robot navigation using a sensor net," in *Proceedings of the Eleventh International Symposium of Robotics Research*. Springer-Verlag, Oct. 2003.

[7] D. K. Arvind and Kai-Juan Wong, "Speckled computing: Disruptive technology for networked information appliances," in *Proceedings of IEEE International Symposium on Consumer Electronics*, Sep. 2004, pp. 219–223.

[8] "Psoc mixed signal array: Technical reference manual," Cypress MicroSystems, 2005.

[9] "Smartrf cc2420 preliminary datasheet," Chipcon, Feb. 2004.

[10] Kai-Juan Wong and D. K. Arvind, "Speckmac: Low-power decentralised mac protocols for low data rate transmissions in specknets," in *Proceedings of the second international workshop on Multi-hop ad hoc networks: from theory to reality*, May 2006, pp. 71–78.

[11] ——, "Perspeckz-64: Physical test-bed for performance evaluation of mac and networking algorithms for specknets," in *Proceedings of the second international workshop on Multi-hop ad hoc networks: from theory to reality*, May 2006, pp. 122–124.

[12] Specksim simulator. [Online]. Available: http://www.specknet.org/dev/specksim