Multi-agent based path planning for first responders among moving obstacles

Zhiyong Wang[a,*], Sisi Zlatanova[a]

[a]*Delft University of Technology, Julianalaan 134, 2628BX Delft, The Netherlands*

Abstract

Natural or man-made disasters can cause different kinds of moving obstacles (e.g., fires, plumes, floods), which make some parts of the road network temporarily unavailable. After such incidents occur, responders have to go to different destinations to perform their tasks in the environment affected by the disaster. Therefore they need a path planner that is capable of dealing with such moving obstacles, as well as generating and coordinating their routes quickly and efficiently. In this paper, we present a novel approach for using multi-agent system for navigating one or multiple responders to one or multiple destinations in the presence of moving obstacles. Our navigation system supports information collection from hazard simulations, spatio-temporal data processing and analysis, connection with a geo-database, and route generation in dynamic environments affected by disasters. We design and develop a set of software geospatial agents that assist emergency actors in dealing with the spatio-temporal data required for emergency navigation, based on their roles in the disaster response. One of the key components of the system is the path planning module, which combines the modified A* algorithm, insertion heuristics, and auction algorithm to calculate obstacle-avoiding routes for multiple responders with multiple destinations. A spatial data model is designed to support the storage of information about the tasks and routes produced during the disaster response. Our system has been validated using four navigation cases. Some preliminary results are presented in this paper and show the potential of the system for solving more navigation cases.

Keywords: Path planning, Multi-agent system, Spatial data model, Algorithms, Moving obstacles

1. Introduction

First responders play an important role in crisis management, saving peoples lives during disasters. They are personnel from different agencies, e.g., fire brigade, police and medical care, and are responsible for a wide range of tasks, including searching for survivors, transporting relief goods, evacuating and transferring wounded people. Most of these emergency tasks require fast and safe navigation as well as the coordination of the response teams. As the economic and human loss due to natural, man-made and human invoked disasters are increasing (Munich RE, 2015), much more research efforts have been devoted to the issues in disaster management and a special attention has been paid to the navigation for first responders in the disasters response.

One of the challenging issues is that natural or man-made disasters can create all sorts of moving obstacles that affect the road network, making some roads dangerous or impossible to traverse. Using traditional routing algorithms, Mioc et al. (2008) and Chitumalla et al. (2008) develop applications that use the forecast information of hazards in the near future in routing and providing navigation services taking blocked areas or streets into account. Nevertheless, they lack a consideration of the dynamics of the environment affected by the moving obstacles, which may make the planned path longer than the shortest one. In some situations, the responders can pass through the threatened roads before they are affected, instead of just avoiding them. Because the status of the road network affected by moving obstacles changes over time, the temporal aspect should also be considered in the routing process. Similar research in the field of robotics on navigation in

*Corresponding author at Architecture and the Built Environment, Julianalaan 134, 2628BX Delft, The Netherlands. Tel.: +31 (0)152787934; fax: +31 (0)152784422.
E-mail addresses: Z.Wang-1@tudelft.nl (Z. Wang), S.Zlatanova@tudelft.nl (S. Zlatanova)

the presence of moving obstacles has been considerable (Phillips & Likhachev, 2011; Li et al., 2009; Narayanan et al., 2012), which could be beneficial to the research on navigation for first responders in some aspects. Another important issue is that the collaborative activities among emergency agencies require the coordination of their routes and destinations. Because the first responders often work in groups and perform tasks together, they need not only to obtain individual routes but also to take into account other response units in the routing process. For example, in the case of emergency medical service, ambulances are distributed to different destinations to pick up and deliver patients according to factors such as the situation of the patients, the deployment of the paramedics, the availability of medical supplies in hospitals, etc. Although numerous techniques have been proposed in logistical planning and robotics to achieve the efficient allocation of vehicles (Dias et al., 2006), they do not have any effective mechanisms to deal with moving obstacles, and can not be applied to the environment affected by disasters. Therefore, there is a great need for an emergency navigation system that is capable of quickly and safely navigating multiple responders to multiple destinations, avoiding moving obstacles.

In this study, we apply the agent technology to address navigation problems with moving obstacles. The method of agent technology was introduced by Wooldridge & Jennings (1995) and represents a diagram for the development of software entities that automates specific computer-based tasks. Agent technology has a set of features, including supporting distributed control, allowing for flexibility and adaptability, etc., which make it suitable for development of the system for navigation in the presence of moving obstacles (Wang & Zlatanova, 2013b). The agent technology has been applied to very varied fields, for example, crisis management (Schoenharl & Madey, 2011), supply chains (Vokřínek et al., 2010; Zeddini et al., 2008). Recently, there has been increasing interest in applying agent technology to GIScience (Sengupta & Sieber, 2007; Crooks & Wise, 2013). Most work in this direction has been devoted to the use of multi-agent systems to simulate and study the behavior and interactions of humans in environments (Chen & Zhan, 2008; Torrens et al., 2012; Bonabeau, 2002). On the other hand, more and more researchers have applied the agent technology to the development of software spatial agents, which assist users in the analysis and evaluation of spatial problems (Genc et al., 2013; Nourjou et al., 2013). However, these agents are usually developed to address a particular set of geospatial issues, which limits their application to other problems, such as emergency navigation. Because the path planning in the presence of moving obstacles is characterized by its reliance on the large amounts of dynamic spatial data produced in disasters (Visser, 2009; Mioc et al., 2008; Wang & Zlatanova, 2013a), special types of agents, combined with GIS functionalities, are needed to quickly process and analyze the spatio-temporal data need for navigation during the disaster response.

In this paper, we focus on addressing a subset of navigation cases presented in Wang & Zlatanova (2013c), and propose an integrated navigation system for first responders in the presence of moving obstacles, based on a multi-agent system. Our system extends the framework described in Wang & Zlatanova (2013b), and introduces a set of modules that are comprised of software agents for its spatial data processing and analysis. Such an approach can allow the system to be easily adjusted and applied to various navigation cases. We use hazard models to provide the predicted information about the obstacles, and select a geo-database in which to store the data needed for emergency navigation. A new one-to-one path planning algorithm is adapted from Wang & Zlatanova (2013a), and is combined with other path planning algorithms to calculate obstacle-avoiding routes for responders with one or multiple destinations. The remainder of the paper is organized as follows: Section 2 presents the navigation cases we aim to address in this paper, and describes the conceptual framework designed for navigation in the presence of moving obstacles. In Section 3, we illustrate the architecture of the proposed multi-agent based navigation system, which contains different types of agents and geo-database. In Section 4, we provide the algorithms used for the different types of path planning problems. Section 5 describes the implementation of our proposed system, and presents some of the results of applying this system to four navigation cases. Finally, we discuss some aspects of the system and conclude with an outline of future research in Section 6.

2. Conceptual analysis and design

This section first provides a taxonomy of navigation in the presence of obstacles. In this taxonomy, we examine and analyse different navigation cases, and place our studies in this paper in the broader context of navigation for first responders. After that, we present our approach, which supports routing in these navigation cases, and describe the general architecture of our navigation system.

2.1. *Conceptual analysis of navigation cases with obstacles*

This research is motivated by navigation problems that arise during disaster response. To help describe the similarities and differences between the navigation cases during disasters, we have constructed a taxonomy in the domain of navigation in the presence of obstacles (Wang & Zlatanova, 2013c), offering some broad keywords and phrases that characterize these cases. In this taxonomy, we identify the following set of criteria and their typical values. We refer to each combination of different values of each criterion as a case, and represent it in the form of a quadruple:

$$< X_1, X_2, X_3, X_4 >$$

where

(1) X_1 is the number of responders (one or many)

(2) X_2 is the number of destinations (One or Many)

(3) X_3 is the type of the destinations (Static or Dynamic)

(4) X_4 is the type of the obstacles (static or moving)

For example, the case denoted by $< o, M, D, m >$ means one moving object has to be routed to many dynamic destinations, avoiding many moving obstacles.

In this taxonomy, there are, in all, sixteen navigation cases. We investigated previous work on these navigation cases in the fields of emergency management and robotics. According to our investigation, only a few studies have paid attention to navigation in the presence of moving obstacles in real road networks (Wang & Zlatanova, 2013a; Visser, 2009). To start with, in this paper we mainly focus on the navigation cases that involve moving obstacles and static destinations. This is because that they occur more often and are simpler than the cases with dynamic destinations. Specifically, we study the four navigation cases as shown in Table 1.

2.2. *Conceptual design of the system architecture*

To address the navigation problems presented above, a conceptual framework for navigating multiple responders in the presence of moving obstacles has been designed. Figure 1 depicts the proposed framework, which combines the following technologies:

- Hazard simulation models. We use hazard simulation models driven by real-time sensor measurements (e.g., wind speed, air temperature, humidity, etc.) to provide predictions of hazards. The data from the hazard models are represented by moving polygons that cross the road network and temporarily close some roads.

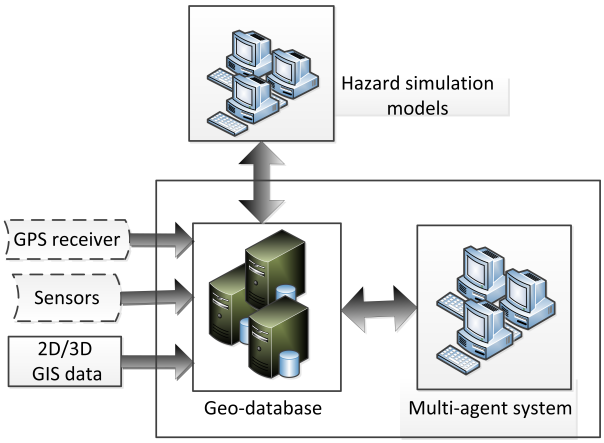


Figure 1: The overview of the generic system architecture

- Geo-database. In our research, a geo-database is selected and serves multiple purposes, including the representation of spatio-temporal information of the road network, the storage of information regarding the emergency tasks and relief vehicles (routes, sources, destinations, travel times, etc.), and supporting interoperability of the proposed system with other crisis management systems, etc.

- Multi-agent system. We extend the work presented in Wang & Zlatanova (2013b), and build a multi-agent system coupled with GIS functionalities to address more navigation cases. The developed multi-agent system supports varied data processing and analysis as well as route calculation.

The calculated routes, along with data about the obstacles and vehicles, are presented to the emergency managers in the control center and forwarded to the mobile devices of the responders in the field as well.

3. The architecture of the multi-agent based navigation system

In this paper, we concentrate on issues related to the multi-agent system and geo-database. In the following sections, we elaborate on the design and development of the agents (the definition of the roles, functionalities, interactions, etc.), routing algorithms, and a spatial data model that structures the data required for the routing and allocation process. Aspects regarding the hazard simulation models are outside the scope of this paper.

The architecture of the multi-agent based navigation system consists of five modules (see Figure 2): 1). Pre-

Table 1: The considered four navigation cases

Navigation case	Description
• $< o, O, S, m >$ *One moving object has to be routed to One Static destination, avoiding many Moving obstacles*	This situation may occur when a relief vehicle has to be navigated through an area affected by toxic plumes. In the case of plumes, the affected roads could be temporarily closed and be available again in the near future. Therefore, a waiting option can be employed in the routing to minimize the total travel time, in the meantime avoiding the moving obstacles
• $< m, O, S, m >$ *Many moving objects have to be routed to One Static destination, avoiding many Moving obstacles.*	A classical example of this situation is navigating a certain number of fire trucks to a fire point. This problem can be split into sub-problems by navigating moving objects separately, which can be addressed by the approaches proposed for $< o, O, S, m >$.
• $< o, M, S, m >$ *One moving object has to be routed to Many Static destinations, avoiding many Moving obstacles.*	One typical example is guiding a fire brigade to several emergency locations in an area affected by floods to rescue victims, clean roads, and help pump water out of the flooded house. The navigation system must be able to plan a trip connecting these locations in a dynamic environment, which can be addressed as a dynamic version of the Traveling Salesman Problem (TSP).
• $< m, M, S, m >$ *Many moving objects have to be routed to Many Static destinations, avoiding many Moving obstacles.*	This may happen when some rescue vehicles are sent to many places to deliver goods and services during a flood event. The navigation problem in this case can be formulated as a Multiple Traveling Salesmen Problem (MTSP), and requires the distribution of the tasks among the responders while taking into account the moving obstacles.

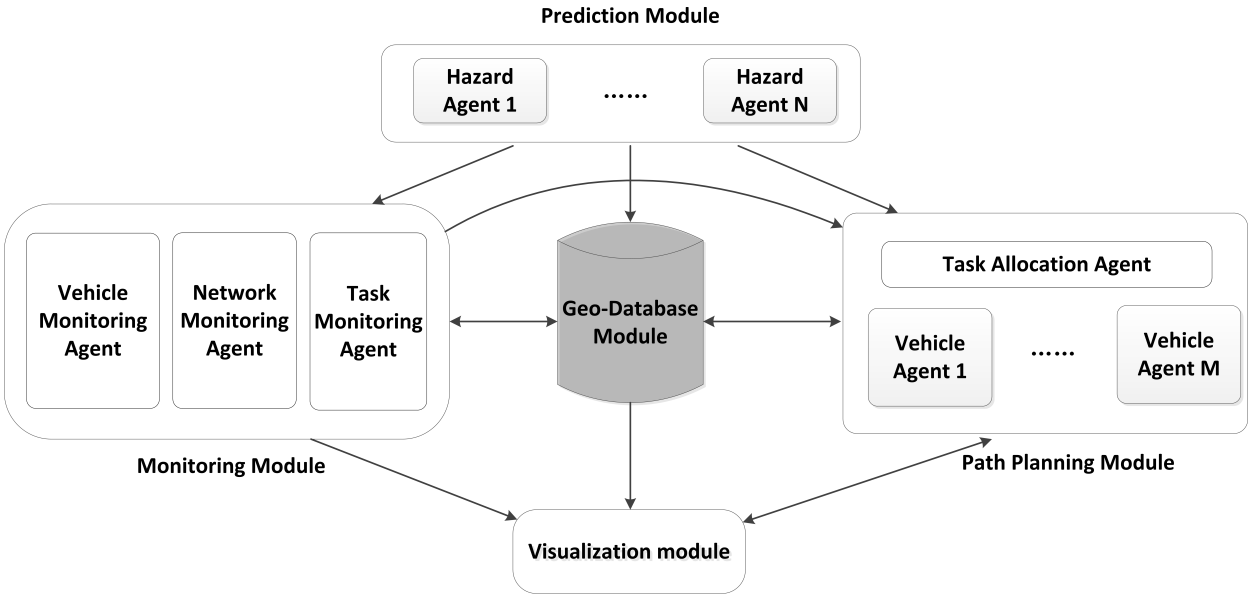


Figure 2: The architecture of the proposed multi-agent based navigation system

diction module; 2). Geo-database module; 3). Monitoring module; 4). Path planning module; 5). Visualization module. The major functions of each module are shown in table 2. In the prediction module, monitoring module, and path planning module, different types of specialized agents are developed to perform the operations involved in emergency navigation and to assist responders in performing their emergency tasks. The geo-database module and the visualization module support the components for the agent system. This multi-agent architecture allows the system to be distributed over different machines, each of which contains one or more types of agents running concurrently. All modules and related agents are presented in more detail in the sections below.

Table 2: The major functions of the modules in the system

Module	Description
Prediction module	Supports the processing of data about the moving obstacles produced from hazard models.
Geo-database module	Organize and store all the information related to navigation.
Monitoring module	Supports the monitoring of the road network, vehicles, and tasks.
Path planning module	Responsible for generating and coordinating the routes.
Visualization module	Provides visualization of the routes as well as the environment.

3.1. Prediction module

The prediction module generates the predicted information about the moving obstacles. This module consists of different types of Hazard Agents, which support handling and operating data from different hazard simulators. To meet the needs of real applications, a Hazard Agent is customized with specific knowledge of hazard simulations and added to the module. For example, in the case of forest fires, a fire agent that can collect data on the fire-affected units from the fire simulation is used, producing the polygons that represent the spread of the fire. Besides, this module also informs the other modules about new predictions generated by the hazard simulations, which allows re-planning the routes if the current plan cannot be carried out due to changes in the situation.

3.2. Geo-database module

An important part of the geo-database is the spatial data model used for structuring the information relevant to the emergency navigation. As described in Table 1, in some navigation cases, responders need to go to multiple destinations to perform their tasks. With these considerations, we provide a data model based on the earlier work (Wang et al., 2014) to handle the situations that involve multiple destinations.

Figure 3 shows a UML diagram of our logical data model for storing the data about the relief vehicles and their routes. The yellow class, RealIncident, is used for storing the data related to incidents. The classes Route (in green) is created to capture the spatial features of the route. The classes Process, Task, and Vehicle (in light-gray) are defined for handling the data related to the emergency response. Newly created datatypes, MovingPolyonInst and MovingPointInst, are colored in purple. The class Task is used in our model to store the destinations of vehicles. It is associated with a Process that manages the RealIncident, and contains the information about the location where this task should be performed. Responders may have different capacities and responsibilities, and need different time for carrying out the tasks, which implies a many-to-many association between Vehicle and Task. To capture this relationship, TaskByVehicle is created by merging the two tables, and is added with an attribute operation_time to store the time required for each task and each vehicle. The operation_time of tasks is updated by the responders in the field and will be used in the routing process. The Vehicle follows the route generated by the path planning algorithms. The class Route contains the information about the routes, which may have one or more tasks along the route. The classTaskInRoute is created for linking Route and Task, and it has an attribute, called seq_id, to indicate the sequence number of the task identified by taskID.

3.3. Monitoring module

The monitoring module makes a direct connection to the geo-database that is updated by the hazard simulations and by the responders in the field, notifies other agents of environment changes in the affected areas, and provides situational information that is further analysed by other modules.

This module consists of the following three types of agents:

- The Task Monitoring Agent constantly checks the state of all tasks stored in the database to see which tasks have been fulfilled or not. Moreover, it also sends the un-completed tasks to the path planning module (see Section 3.4) for the planning or re-planning of routes, if a route calculation request is received.

- The Network Monitoring Agent has two intentions: 1). collect spatio-temporal information about the road network, using the predicted data of the hazards. It performs intersection operations between the obstacle polygons and the roads, determining all the affected roads and the time intervals when they will be closed or open; 2). Make requests for re-planning routes.

- The Vehicle Monitoring Agent performs real-time tracking of the moving vehicles. It uses GPS data to analyse the state of the moving vehicles. If a vehicle is found to be trapped in a traffic jam, or to be unable to fulfill its task, the Vehicle Monitoring Agent will send a list of all available vehicles to the path planning module, and suggest a re-planning

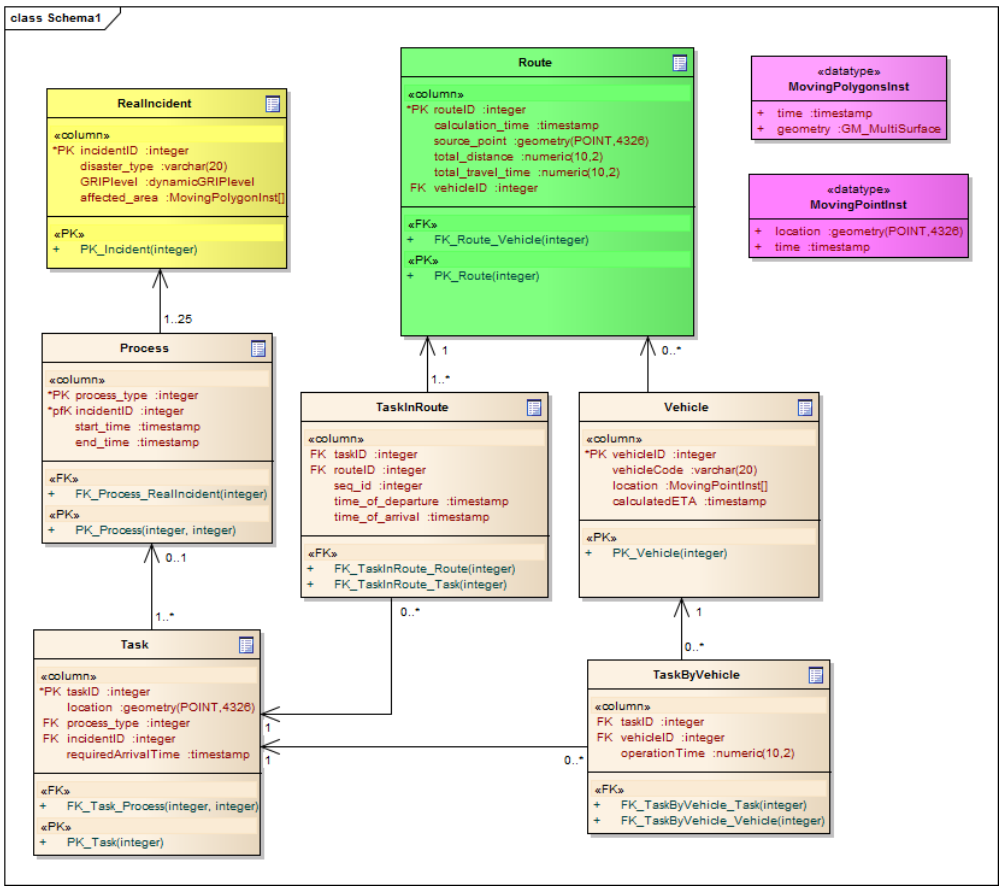


Figure 3: UML diagram of the logical data model for managing information relevant to relief vehicles

of the routes for the other vehicles to fulfill these tasks.

3.4. Path planning module

This path planning module is the core of the system. It makes a central plan for the routes and broadcasts this to the vehicles. The routes are generated by means of a distributed mechanism implemented by the MAS, which has two types of agent: Task Allocation Agent and Vehicle Agent. When a route calculation is requested, the Task Allocation Agent first communicates with the Task Monitoring Agent for tasks that have not been fulfilled, and obtains the information about the available vehicles from the Vehicle Monitoring Agent. Then it allocates tasks to the Vehicle Agents. A Vehicle Agent is associated with its corresponding rescue unit (e.g., a fire truck, a police car, and an ambulance), and uses the routing algorithms (see Section 4 to calculate the cost of carrying out the tasks, and to bid on the new tasks. The bid, together with the calculated results of the routes (e.g., travel time, completion time of task, travel distance, etc.), is forwarded to the Task Allocation Agent for evaluation.

3.5. Visualization module

The visualization module has been developed to show how the path would avoid the predicted obstacles and help the responders to decide whether they will use the proposed path or not. This module can run on local desktop computers. It allows users to check the predicted information for moving obstacles, select destinations for the vehicles, and evaluate the planned routes. We have developed both 2D and 3D viewers in this module. But for the purpose of this paper, only the 2D visualization will be presented here. This 2D viewer is built using OpenStreetMap as a base map. It visualizes the regions where the first responders work and provides an overview of the planned paths. More importantly, it also supports the animation of the predicted movement of the hazards and vehicles. The viewer can display the

movement of the obstacles that cross the road network, and the activities of the agents that represent the responders who move and carry out their tasks, by means of which the situational awareness of the emergency managers can be enhanced.

4. Algorithms used for route determination in the path planning module

Basically, the algorithms calculate optimal or near-optimal routes for three types of path panning problems: one-to-one, one-to-many, and many-to-many, considering multiple parameters (the time required to carry out the task, the speeds of the vehicles, and the departure times). The details of the algorithms follow.

4.1. One-to-one path planning

Let $G = (N, E)$ be a network consisting of a finite set of nodes N and edges E. We use l_e to represent the length of each edge e. The edge between two nodes x and y is denoted by xy. To represent the changes in the availability of the roads, each edge in the road network is associated with a set of temporal intervals, $S_e = (p_1, ..., p_i, ..., p_I)$, $e \in E$, to represent the its available state. $p_i = [t_{oi}, t_{ci}], t_{oi} < t_{ci}$, indicates the time period in which the edge is accessible, where t_{oi} is the start time of the opening, t_{ci} denotes the end time of the opening, and I is the total number of openings of this edge. We also use $S_x = (p_1, ..., p_j, ..., p_J)$, $x \in N$ to represent and store the state of the node. All this dynamic information for the road network will be handled by our path planning algorithm to calculate routes that avoid the moving obstacles.

In our system, we develop a new algorithm to solve the one-to-one path planning problem with moving obstacles, aiming to minimize the total travel time. The developed algorithm has some similarities to the algorithms that are developed by Visser (2009) and Wang & Zlatanova (2013a), but our algorithm can be applied to more complex situations, and is able to deal with roads that have more than two blocks caused by moving obstacles. Using the concept of safe intervals from the works of Phillips & Likhachev (2011) and Narayanan et al. (2012), we extend the A* algorithm to calculate the obstacle avoiding routes in the road network. In the extensions of A*, we take into account the safe intervals of the edges and the *speed* of vehicles in the generation of successors of a state. The *departure_time* is introduced to estimate the arrival of each state. Figure 4 shows an outline of our extended algorithm. In the algorithm, the state s of node x is represented by (x, p_j), where $p_j = [t^x_{oj}, t^x_{cj}]$ is the jth safe interval of node x. In addition to the variables of the traditional A*, we introduce $w(s, s')$ to store the waiting time needed for moving from one state s to another state s'. When a state s is expanded, we generate its successors using the function $UpdateOpenSet(s)$ (see Figure 5), and insert them into the $openSet$ for further expansion. We estimate whether the vehicle can pass through the edge within each safe interval $[t^{xy}_o, t^{xy}_c]$ of edge xy, considering the safe interval of node x (line 3-9 in Figure 5). If node y can be reached within a given safe interval of xy, we generate a new state s' for node y, and compute the earliest arrival time of s' considering the length and the speed. The vehicle can take a waiting action at node x in order to pass through the edge safely. The waiting time $w(s, s')$ is calculated based on the difference between the earliest start time from s to s' and the arrival time of state s, i.e., $g(s)$. The generated state s' is updated and inserted into $openSet$ for further expansion.

The modified A* algorithm

```
1: initialize source, target, speed, departure_time
2: openSet = ∅, closedSet = ∅
3: for each interval j of node source do
4:    if (t_oj^source < departure_time) and (departure_time < t_cj^source) then
5:       generate state s_source = (source, p_j)
6:       g(s_source) = departure_time
7:       break
8:    end if
9: end for
10: insert s_source in openSet
11: while openSet is not empty do
12:    s = the state in openSet having the lowest f value
13:    if node(s) = target then
14:       return the path from source to target
15:    end if
16:    remove s from openSet
17:    insert s to closedSet
18:    updateOpenSet(s)
19: end while
20: return no-path
```

Figure 4: The modified A* algorithm

4.2. One-to-many path planning

In this study, we use the insertion heuristic to find near optimal solutions to one-to- many path planning problems. Using simple but effective rules, insertion heuristics have been applied to many variants of the Travelling Salesman Problem (TSP), and can produce quality solutions quickly. Figure 6 describes our one-to-many path planning algorithm based on the insertion method. The objective of the algorithm is to minimize

```
updateOpenSet(s)
1: s = (x, p_k), x = node(s), p_k = [t^x_ok, t^x_ck]
2: for each neighbor y of x do
3:   for each interval i of edge xy do
4:     if (t^x_ck < t^xy_oi) or (t^xy_ci < g(s)) then
5:       continue
6:     else
7:       start_time = max(g(s), t^xy_oi)
8:       tentative_cost = start_time + l_xy/speed
9:       if tentative_cost < t^xy_ci then
10:        for each interval j of node y do
11:          if (t^y_oj < tentative_cost) and (tentative_cost < t^y_cj) then
12:            generate new state s' = (y, p_j)
13:            w(s, s') = start_time - g(s)
14:          end if
15:        end for
16:        if s' is in closedSet then
17:          continue
18:        end if
19:        if s' is in openSet then
20:          if tentative_cost < g(s') then
21:            g(s') = tentative_cost
22:            f(s') = g(s') + h(s')
23:          end if
24:        else
25:          g(s') = tentative_cost
26:          f(s') = g(s') + h(s')
27:          insert s' into openSet
28:        end if
29:      end if
30:    end if
31:  end for
32: end for
```

Figure 5: UpdateOpenSet(s)

the total trip duration, which includes the total traveling time and the operation time of the tasks. Given a set of k tasks, $T = \{T_1, \ldots, T_i, \ldots, T_k\}$, $1 <= i <= k$, that need to be allocated to a responder. Each task corresponds to an instance of class Task in the data model. The algorithm calculates the path cost of the insertion of the evaluated task T_i in each position. Then it selects the position that makes the cost of the new path be minimal. Finally it chooses the task that has the maximum path cost to ensure that the worst task is inserted into the path P first. Each trip between two tasks in the path is calculated by the modified A* algorithm presented in Section 4.1 , based on the given *speed* as well as its departure time. Because the responders should spend a certain amount of time in performing their tasks, the operation time of each task is also important in the path planning among moving obstacles. In our system, the operation time is extracted from the operation_time of TaskByVehicle in the database, and added into the algorithm to derive the time of travel to the next task as follows:

$$t_{\mathrm{D}}(T_j) = t_{\mathrm{A}}(T_j) + t_{\mathrm{O}}(T_j) \tag{1}$$

Here, $t_{\mathrm{D}}(T_j)$ is the departure time from the position of task T_j, $t_{\mathrm{A}}(T_j)$ is the arrival time to the position of task T_j, and $t_{\mathrm{O}}(T_j)$ is the operation time of task T_j.

In many crisis situations, emergency tasks require a specific time for the arrivals, and the responders have to reach the destinations in a limited amount of time. On the other hand, the influence of moving obstacles on the roads can cause longer trips for the vehicles, and result in delays beyond the required time for the responders. Therefore, we use the completion time of the tasks as one of criteria in selecting the routes that responders should follow, taking into account the required arrival time for the tasks and the delays caused by moving obstacles. In this study, we provide a multi-objective function that tries to minimize the number of delays, the sum of all the delays, and the task-completion time: Let $b_{\mathrm{delay}}(T_j)$ be a binary variable that indicates whether the vehicle arrives later than the required arrival time of task T_j, and is defined as follows:

$$b_{\mathrm{delay}}(T_j) = \begin{cases} 1 & \text{if } t_{\mathrm{A}}(T_j) > t_{\mathrm{RA}}(T_j) \\ 0 & \text{if } t_{\mathrm{A}}(T_j) \leq t_{\mathrm{RA}}(T_j) \end{cases} \tag{2}$$

where $t_{\mathrm{A}}(T_j)$ is the actual arrival time, and $t_{\mathrm{RA}}(T_j)$ is the required arrival time for task j. Let $t_{\mathrm{delay}}(T_j)$ denote the time delay between $t_{\mathrm{A}}(T_j)$ and $t_{\mathrm{RA}}(T_j)$ for task j. Then it can be written as follows:

$$t_{\mathrm{delay}}(T_j) = \begin{cases} t_{\mathrm{A}}(T_j) - t_{\mathrm{RA}}(T_j) & \text{if } t_{\mathrm{A}}(T_j) > t_{\mathrm{RA}}(T_j) \\ 0 & \text{if } t_{\mathrm{A}}(T_j) \leq t_{\mathrm{RA}}(T_j) \end{cases} \tag{3}$$

Using above notation, we formulate the following multi-objective function for minimization of the path cost of a vehicle:

$$F(v, P) = M(v, P) + \alpha \times N(v, P) + \beta \times D(v, P) \tag{4}$$

where $F(v, P)$ is the cost of the entire path P of vehicle v, $M(v, P)$ is the completion time of all tasks in path P (in minutes), $N(v, P)$ represents the number of delays, $\sum_{j=1}^{k} b_{\mathrm{delay}}(T_j)$, $D(v, P)$ corresponds to the sum of all time delays (in minutes), $\sum_{j=1}^{k} t_{\mathrm{delay}}(T_j)$, and α, β are penalty weights.

The weights α and β are set based on the priority of the corresponding objective in the function: an objective with a higher priority has a higher weight. In our research, we try to minimize the number of delays first $N(v, P)$, next minimize the sum of all delays $D(v, P)$, and then the completion time of tasks $M(v, P)$. Given these priorities, we set the penalty weights in the following way: $\alpha > \beta > 1$.

Incremental insertion method

```
source, departure_time, speed
T = {T_1, T_2, ..., T_k}
P = ∅ ; // the sequence of tasks forming the path
while T ≠ ∅ do
  for i = 1 → |T| do
    select task T_i from T
    if (|P| = 0) then
      calculate the cost of the path from source to T_i
    else
      for j = 1 → |P| + 1 do
        insert task T_i in the position j of P
        calculate the path from source to T_{I_1}
        for l = 2 → |P| + 1 do
          calculate the departure time from
              the position of T_{I_{l-1}} by formula (1)
          calculate the path from T_{I_{l-1}} to T_{I_l}
        end for
        calculate the cost of the new path P by formula (4)
      end for
      insert T_i in the position that produces the minimum path cost
    end if
  end for
  select T_max with the maximum path cost
  P = P ∪ {T_max}
  T = T\{T_max}
end while
```

Figure 6: Incremental insertion method (adapted from Skiena (2008))

4.3. Many-to-many path planning

Considering computational efficiency in the multi-agent coordination (Schoenig & Pagnucco, 2011), in this study we apply sequential single-item auctions (SSI) (Koenig et al., 2006) for allocating the tasks to the vehicles. Figure 7 shows the main structure of the SSI auctions. During an auction, the task allocation agent first makes an announcement to all vehicle agents notifying them about new tasks. Then the vehicle agents estimate the cost of completing the unallocated tasks and the already assigned tasks, using the one-to-many path algorithm presented in Section 4.2, and submit bids with the calculated cost. We use the minmax team objective (Lagoudakis et al., 2005) to minimize the maximum path cost over all vehicles, i.e., $\min \max_{i=1}^{|V|} F(v_i, P_i)$. For this objective, each vehicle agent should bid with its cost $c = F(v_i, P_i)$ on target t. Finally the task allocation agent collects the bids and determines the winner of the auction, assigning a single task to the vehicle agent with the lowest cost. The above described process is repeated until all tasks are assigned. Because the predictions from the hazard simulation change with real sensor measurements, the previously generated allocation plans might not be applicable to the new situations, but rather need to be adapted. In our research, we use a dynamic method for re-allocating tasks. This method repeats the auctions while the tasks are being executed. When new predicted information of hazards is gathered, the task allocation is restarted and all unaccomplished tasks will be auctioned again. This enables the system to dynamically allocate the tasks in the environment affected by the disasters, providing the ability to adapt to changing conditions.

Sequential single-item (SSI) auction

```
T = {T_1, T_2, ..., T_k}
T(v) = ∅
while T ≠ ∅ do
  for each task t ∈ T do
    for each vehicle agent v ∈ V do
      estimate the cost c of carrying out tasks T(v) ∪ {t}
      send bid(c, t) to the task allocation agent
    end for
  end for
  the task allocation agent
    selects the v_s with the smallest bid (c_s, t_s)
  T = T\{t_s}
  T(v_s) = T(v_s) ∪ {t_s}
end while
```

Figure 7: Sequential single-item (SSI) auction (adapted from Koenig et al. (2006))

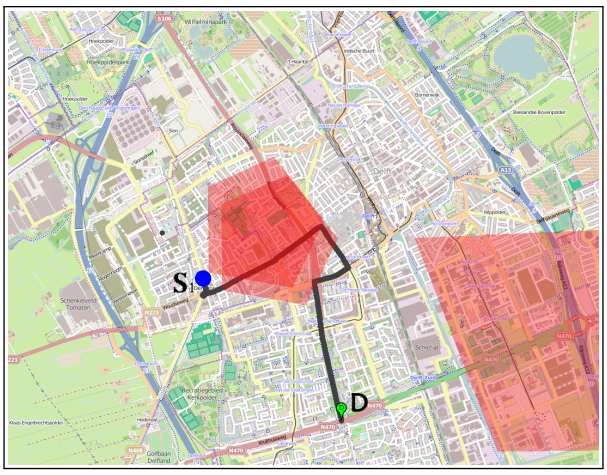
5. Implementation and case studies

Following the structure of the system presented in the previous sections, a prototype of a multi-agent based navigation system has been implemented. In our system, we use JADE (Java Agent Development Framework) (Bellifemine et al., 2005) as the underlying agent infrastructure, and combine it with another agent-based toolkit, GeoMASON (Sullivan et al., 2010). GeoMASON is a GIS extension of the simulation toolkit Mason (Luke et al., 2004). The proposed spatial data model is realized in the relational database PostGIS (www.postgis.org) to store the needed data. The calculated routes, together with the data about the moving obstacles and relief vehicles, are delivered to GeoMason to configure the visualization module, and are displayed to users through the developed 2D viewer.

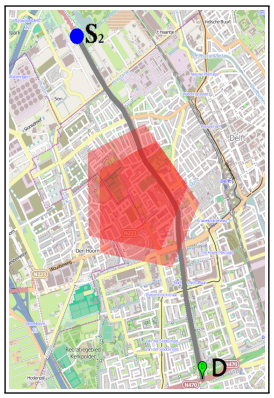
To assess the capability of our navigation system in terms of calculating obstacle-avoiding routes, the system and algorithm have been tested with the road network dataset in Delft, the Netherlands. The network is composed of 1586 edges and 1780 nodes. We consider the following specific crisis response scenario. Suppose that a poisonous material has been accidentally released into the city and some plumes affect the central part of

the city. The first responders are distributed to perform a number of rescue tasks, for instance, assisting injured people who require medical service, taking measurements and observations, and evacuating refugees from the dangerous areas. We apply the system to the four navigation cases presented in Section 2.1. The calculated results are as follows:

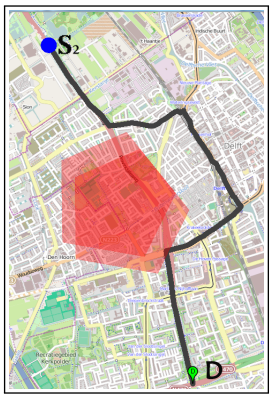
5.1. *Case 1: $< o, O, S, m >$ navigation of one responder to one destination*



(a) The shortest route R0 and the route R1 calculated by the modified algorithm for vehicle v_1 from source S_1 to destination D (R1 has the same geometry as R0)



(b) The shortest route R2 for vehicle v_2 from source S_2 to destination D



(c) The route R4 calculated by the modified A* algorithm for vehicle v_2 from source S_2 to destination D

Figure 8: Snapshot of the calculated routes and the moving obstacles (in red polygons)

In this case, the emergency manager needs to select a vehicle to go to a destination, meeting the requirements on the response time. Here we assume that the responders should reach the destination in 10 min. There are two vehicles v_1 and v_2 that are available and can move at a speed of 40 km/h. The system computes the obstacle-avoiding routes for these two vehicles. For comparison purposes, we also use the standard A* algorithm to calculate the shortest routes. Table 3 shows the calculated results. Figure 8 displays the routes for each vehicle on the map. The light-grey line is the shortest route, and the dark line is the route calculated by our algorithm considering the speed of 40 km/h. As shown in the table, although v_1 is the nearest vehicle to the destination, it has to wait for a long time to avoid obstacles according to the estimation provided by our algorithm. This results in vehicle v_1's failing to arrive within the required response time. On the other hand, vehicle v_2, following the calculated route, can reach the destination before the expected arrival time without any waiting. Because the moving obstacle could cause possible delays during the response, these delays should also be considered during crisis decision making. Taking into account the moving obstacles, the system can not only provide safe and fast routes, but also support emergency managers in the selection of response teams to handle the incidents within the required time limit.

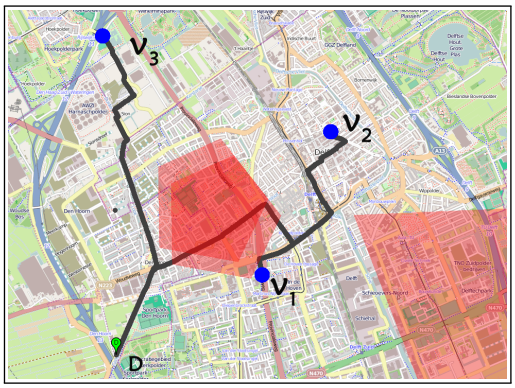
Table 3: Calculated results of the case $< o, O, S, m >$

Vehicle ID	Route ID	Distance (km)	Total waiting time (min)	Arrival time (min)
v_1	R0	4.16	X	X
	R1	4.16	10.7	16.9
v_2	R2	4.23	X	X
	R3	5.77	0	8.5

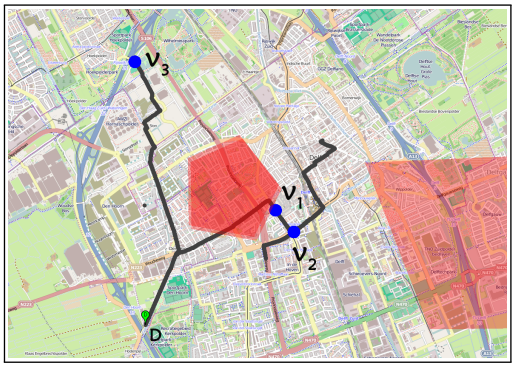
Notes:
[1] The vehicles considered in this scenario departure at time t = 0 min
[2] R0, R2: The shortest route calculated by the standard A* algorithm
[3] R1, R3: The route calculated by the modified A* algorithm at a speed of 40 km/h (the distance of R1 equals the distance of R0)
[4] X: no value

5.2. *Case 2: $< m, O, S, m >$ navigation of multiple responders to one destination*

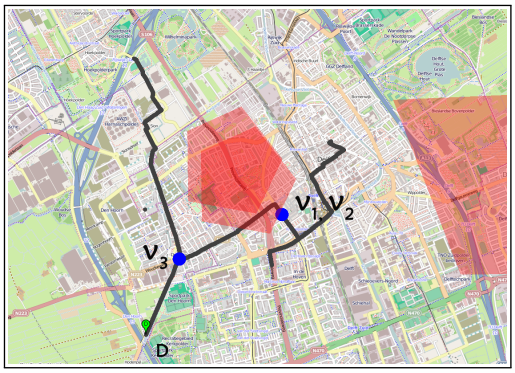
Table 4 shows the results of applying the system to a many-to-one path planning example. In our system, each vehicle reports their profile, such as the current position, speed, and departure time, and the system calculates the obstacle-avoiding path for each vehicle. As we can see from the table, only vehicle v_3 can arrive at the destination on time: v_1 has to spend 9.7 min more



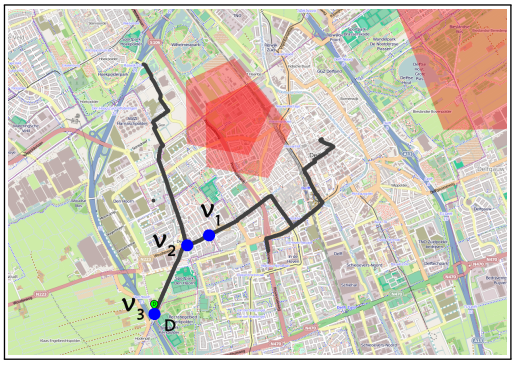
(a) t = 0 min, v_1 , v_2 v_3 are at the different source points



(b) t = 4 min, v_1 is waiting to avoid moving obstacles, v_2 is moving along the calculated route, and v_3 is still at the source point



(c) t = 8 min, v_1 and v_2 are waiting at the same location to avoid moving obstacles, and v_3 is moving towards the destination



(d) t = 16 min, v_1 and v_2 continue moving towards the destination, and v_3 reaches the destination

Figure 9: Snapshots of movements of both the moving obstacles (in red polygons) and the vehicles (in circle) at different times

Table 4: Calculated results of the case $< m, O, S, m >$

Vehicle ID	Route ID	Departure time (min)	Total travel time (min)	Arrival time (min)
v_1	R0	0.0	9.1	9.1
	R1	0.0	18.8	18.8
v_2	R2	1.0	7.8	8.8
	R3	1.0	16.5	17.5
v_3	R4	4.0	5.4	9.4
	R5	4.0	5.4	9.4

Notes:
[1] R0, R2, R4: The shortest routes from different sources to the same destination
[2] R1: The route calculated by the modified A* algorithm given a speed of 30 km/h
[3] R3, R5: The route calculated by the modified A* algorithm given a speed of 40 km/h (the route R4 and the shortest route R5 are the same)

traveling time to avoid obstacles, which causes the actual arrival time to be later than the arrival time estimated by the shortest route, and vehicle v_2 also has a time delay of about 8.7 min. Figure 9 shows snapshots of the movements of the involved vehicles towards the same destination. The results reveal that the incorporation of predicted data of hazards is essential for the estimation of the arrival times of the relief vehicles, and thus contributes to the generation of better emergency plans.

5.3. *Case 3: $< o, M, S, m >$ navigation of one responder to multiple destinations*

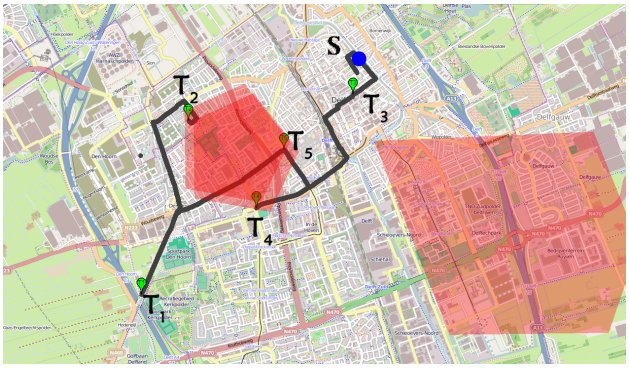


Figure 10: Snapshot of calculated routes for vehicle v_1 (t = 0 min, the vehicle is at the source point, and the moving obstacles are represented by the red polygons)

Figures 10 and 11 depict two scenarios in which five tasks have to be allocated to two vehicles. The tasks of the same ID have the same locations, but differ in operation time needed for responders to perform. We

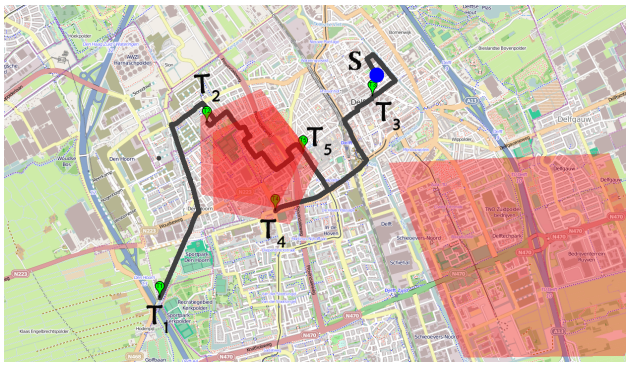


Figure 11: Snapshot of calculated routes for vehicle v_2 ($t = 0$ min, the vehicle is at the source point, and the moving obstacles are represented by the red polygons)

Table 5: The operation time of tasks (min)

v_1	Task	T_1	T_2	T_3	T_4	T_5
	Operation time	1	5	1	5	3
v_2	Task	T_1	T_2	T_3	T_4	T_5
	Operation time	2	3	1	4	5

assume that the vehicles move at a constant speed of 30 km/h and no tasks have a limitation on arrival time. The operation time of the tasks for each vehicle is listed in Table 5. The proposed navigation system plans a trip connecting the locations associated with the involved tasks, considering both the operation time of the task and the predicted information about the environment affected by the plumes. The directions on the routes for the two vehicles are as follows: 1). vehicle $v_1 : S \rightarrow T_3 \rightarrow T_4 \rightarrow T_5 \rightarrow T_1 \rightarrow T_2$; 2). vehicle $v_2 : S \rightarrow T_3 \rightarrow T_4 \rightarrow T_5 \rightarrow T_2 \rightarrow T_1$. As we can see from the results, the vehicles get different routes customized based on their operation time for tasks. Besides, the results also imply the importance of including of the operation time in determining the sequence of tasks that should be followed by the responder, and in the generation of routes that are to avoid moving obstacles.

5.4. *Case 4: $< m, M, S, m >$ navigation of multiple responders to multiple destinations*

Figures 12 and 13 depict routes for the same situation, involving three relief vehicles and 7 assistance tasks, $|V| = 3$, $k = 7$, but differ in the penalty weights used in the calculation. We assume the involved vehicles move at the same speed of 40 km/h, and the operation time of all tasks is 1 min. Our system calculates routes for the involved vehicles by using the algorithm presented in Section 4.3. As previously mentioned, minimizing the number of delays is considered as our

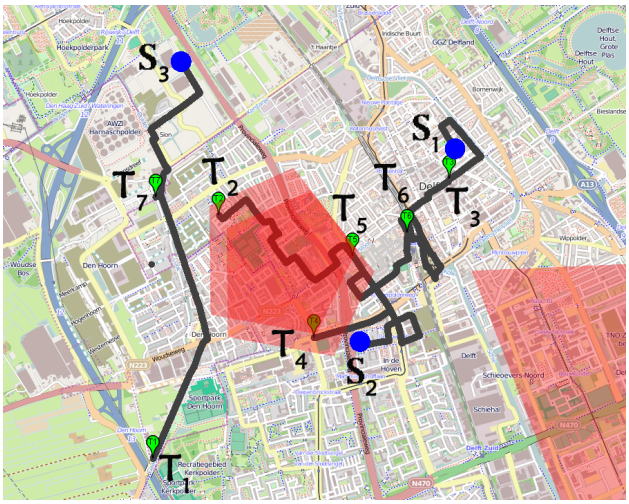


Figure 12: Snapshot of calculated routes for 3 vehicles with 7 tasks ($\alpha = 100000$, $\beta = 1000$, $t = 0$ min, all vehicles are at the source points, and the moving obstacles are represented by the red polygons)

Table 6: Calculated results using the penalty weights ($\alpha = 100000$, $\beta = 1000$)

Vehicle ID	v_1			v_2		v_3	
Task	T_3	T_6	T_2	T_4	T_5	T_7	T_1
t_A (min)	5.6	9.4	17.3	6.2	13.6	5.2	9.4
t_{RA} (min)	20	16	11	8	9	12	8
t_{delay} (min)	0	0	6.3	0	4.6	0	1.4

primary objective, which is then followed by minimizing the total amount of time in the delays. Thus, for the multi-objective optimization of path P_i of vehicle v_i, the largest penalty weight should be imposed on $N(v_i, P_i)$ in order to direct the search towards solutions with lesser numbers of delays. In this scenario, we used the following weights: $\alpha = 10000$, $\beta = 1000$. Table 6 shows the calculated results. The allocated results are as follows: 1) path P_1 of vehicle v_1: $S_1 \rightarrow T_3 \rightarrow T_6 \rightarrow T_2$; 2) path P_2 of vehicle v_2: $S_2 \rightarrow T_4 \rightarrow T_5$; 3) path P_3 of vehicle v_3: $S_3 \rightarrow T_7 \rightarrow T_1$. We also applied our algorithm to this case without using penalty weights ($\alpha = 0$, $\beta = 0$). The allocated results are as follows (see Table 7): 1) path P_1 of vehicle v_1: $S_1 \rightarrow T_3 \rightarrow T_6$; 2) path P_2 of vehicle v_2: $S_2 \rightarrow T_4 \rightarrow T_5$; 3) path P_3 of vehicle v_3: $S_3 \rightarrow T_7 \rightarrow T_1 \rightarrow T_2$. From these tables, we can see that the maximum delay, $\max_{i=1}^{|V|} D(v_i, P_i) = 6.3$ min, in Table 6 is slightly higher than $\max_{i=1}^{|V|} D(v_i, P_i) = 4.9$ min in Table 7. However, by applying the penalty weights, the max number of delays $\max_{i=1}^{|V|} N(v_i, P_i)$ is reduced from 2 to 1.

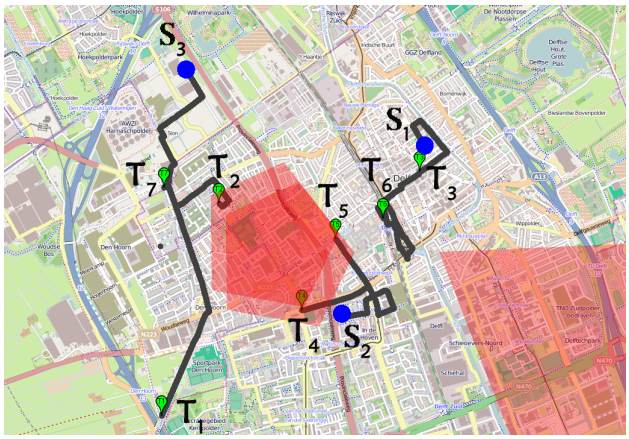


Figure 13: Snapshot of calculated routes for 3 vehicles with 7 tasks ($\alpha = 0, \beta = 0, t = 0$ min, all vehicles are at the source points, and the moving obstacles are represented by the red polygons)

Table 7: Calculated results without using the penalty weights ($\alpha = 0$, $\beta = 0$)

Vehicle ID	v_1		v_2		v_3		
Task	T_3	T_6	T_4	T_5	T_7	T_1	T_2
t_A (min)	5.6	9.4	6.2	13.6	5.2	9.4	14.5
t_{RA} (min)	20	16	8	9	12	8	11
t_{delay} (min)	0	0	0	4.6	0	1.4	3.5

6. Conclusions and future research

In this paper, we presented a multi-agent based navigation system, with the aim of providing solutions to different types of navigation problems. The proposed system integrates hazard models, agent technology, GIS technology, a geo-database, and routing algorithms. We used hazard simulations to provide spatial-temporal information about moving obstacles. A set of software agents was designed and developed to support the spatial data processing and analysis involved in the routing process. To support the path planning and evaluation, a spatial data model was defined to structure the information about the emergency tasks and relief routes in the geo-database. We extended the A* algorithm to calculate one-to-one paths that avoided moving obstacles, used insertion heuristics for one-to-many path planning, and applied SSI auctions to many-to-many path planning problems. The results of the application of the system to different navigation cases demonstrated its ability of generate obstacle-avoiding routes for one or multiple responders to one or many destinations.

Although the system has shown its capabilities to support navigation in the presence of moving obstacles, there are still some limitations that may affect the usefulness of the proposed system in real disaster situations, which need to be considered in future developments. First, a major concern is the accuracy of the input data, such as simulated data from hazard models, the estimated operation times of the emergency tasks, the real time positions of the relief vehicles, etc. The uncertainties and errors that arise from collecting and generating these data play a special role in the determination of the routes. Therefore, an integration of the uncertainties in the data would be necessary to make the calculated routes more reliable and feasible. Second, we currently assume that the drivers follow the calculated routes, and the present developed system doesn't consider the involvement of the vehicle drivers. Because the drivers can make their own decisions on the choice of routes, the system should be able to take the drivers' behavior into account, and to adjust the planned routes according to the actual situations. Third, because the obstacles change as the route calculation is being performed, the time for processing and analysing data is also very important for real-time routing. For one thing, the hazard simulation could produce a large amount of information about hazards which requires considerable computational efforts and time. Thus more agents would be needed in the system to process the data as quickly as possible. For another, some responders may be responsible for certain specific areas, which could be used to facilitate the route calculation by limiting the size of the road network.

In our future research, several extensions will be studied to enhance the routing capability of the system. To begin with, we will explore the use of semantics to aid navigation. Not only information about the hazards (e.g., floods, fires), but also the information about the users (e.g., truck, jeep) will be considered in the routing. Users with different profiles may have different objectives for optimization of the path planning problems, such as maximizing the safety, minimizing the total risk, etc. Currently the values of the penalty weights are chosen based on a preliminary experimentation with the algorithms. Some modifications of the existing algorithms will be developed for adjusting the penalty weights to deal with these objectives. Moreover, because the communication infrastructure may not be available or work properly during a disaster response, a decentralized method is needed to allow different users to negotiate with each other and to make local agreements on the distribution of tasks in case there is no support from the central planning system. Another type of multi-agent system should also be designed to handle this situation. Besides, as presented in Wang & Zlatanova (2013c), there are still a couple of navigation cases that need to be addressed, especially the ones that involve dynam-

ic destinations. More algorithms would be needed to solve these navigation problems. Last but not the least, because traffic situations in the road network, such as car crash, congestion and large traffic flow, have a big impact on the movement of relieve vehicles, the traffic information should also be considered in the path planning in the presence of moving obstacles .

References

References

Bellifemine, F., Bergenti, F., Caire, G., & Poggi, A. (2005). Jade – a java agent development framework. In *Multi-Agent Programming* (pp. 125–147). Springer.

Bonabeau, E. (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, *99*, 7280–7287.

Chen, X., & Zhan, F. B. (2008). Agent-based modelling and simulation of urban evacuation: relative effectiveness of simultaneous and staged evacuation strategies. *Journal of the Operational Research Society*, *59*, 25–33.

Chitumalla, P. K., Harris, D., Thuraisingham, B., & Khan, L. (2008). Emergency response applications: Dynamic plume modeling and real-time routing. *IEEE Internet Computing*, *12*, 38–44.

Crooks, A. T., & Wise, S. (2013). Gis and agent-based models for humanitarian assistance. *Computers, Environment and Urban Systems*, *41*, 100–111.

Dias, M. B., Zlot, R., Kalra, N., & Stentz, A. (2006). Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, *94*, 1257–1270.

Genc, Z., Heidari, F., Oey, M. A., van Splunter, S., & Brazier, F. M. (2013). Agent-based information infrastructure for disaster management. In *Intelligent Systems for Crisis Management* (pp. 349–355). Springer.

Koenig, S., Tovey, C., Lagoudakis, M., Markakis, V., Kempe, D., Keskinocak, P., Kleywegt, A., Meyerson, A., & Jain, S. (2006). The power of sequential single-item auctions for agent coordination. In *Proceedings of the National Conference on Artificial Intelligence* (p. 1625). volume 21.

Lagoudakis, M. G., Markakis, E., Kempe, D., Keskinocak, P., Kleywegt, A. J., Koenig, S., Tovey, C. A., Meyerson, A., & Jain, S. (2005). Auction-based multi-robot routing. In *Robotics: Science and Systems*. volume 5.

Li, H., Yang, S. X., & Seto, M. L. (2009). Neural-network-based path planning for a multirobot system with moving obstacles. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, *39*, 410–419.

Luke, S., Cioffi-Revilla, C., Panait, L., & Sullivan, K. (2004). Mason: A new multi-agent simulation toolkit. In *Proceedings of the 2004 SwarmFest Workshop*. volume 8.

Mioc, D., Anton, F., & Liang, G. (2008). On-line street network analysis for flood evacuation planning. In *Remote Sensing and GIS Technologies for Monitoring and Prediction of Disasters* (pp. 219–242). Springer.

Munich RE (2015). *Topics Geo: Natural Catastrophes 2014: Analyses, Assessments, Positions*. Munchener Ruckversicherungs-Gesellschaft.

Narayanan, V., Phillips, M., & Likhachev, M. (2012). Anytime safe interval path planning for dynamic environments. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 4708–4715).

Nourjou, R., Hatayama, M., Smith, S. F., Sadeghi, A., & Szekely, P. (2013). Design of a GIS-based assistant software agent for the incident commander to coordinate emergency response operations. *In Workshop on Robots and Sensors integration in future rescue INformation system (ROSIN' 13)*, .

Phillips, M., & Likhachev, M. (2011). SIPP: Safe interval path planning for dynamic environments. In *2011 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 5628–5635).

Schoenharl, T., & Madey, G. (2011). Design and implementation of an agent-based simulation for emergency response and crisis management. *Journal of Algorithms & Computational Technology*, *5*, 601–622.

Schoenig, A., & Pagnucco, M. (2011). Evaluating sequential single-item auctions for dynamic task allocation. In *AI 2010: Advances in Artificial Intelligence* (pp. 506–515). Springer.

Sengupta, R., & Sieber, R. (2007). Geospatial agents, agents everywhere... *Transactions in GIS*, *11*, 483–506.

Skiena, S. S. (2008). *The Algorithm Design Manual*. Springer.

Sullivan, K., Coletti, M., & Luke, S. (2010). Geomason: Geospatial support for mason. *Department of Computer Science, George Mason University, Technical Report Series*, .

Torrens, P. M., Nara, A., Li, X., Zhu, H., Griffin, W. A., & Brown, S. B. (2012). An extensible simulation environment and movement metrics for testing walking behavior in agent-based models. *Computers, Environment and Urban Systems*, *36*, 1–17.

Visser, I. (2009). *Route determination in disaster areas*. Master's thesis Utrecht University, Netherlands.

Vokřínek, J., Komenda, A., & Pěchouček, M. (2010). Agents towards vehicle routing problems. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems* (pp. 773–780).

Wang, Z., & Zlatanova, S. (2013a). An A*-based search approach for navigation among moving obstacles. In *Intelligent Systems for Crisis Management* (pp. 17–30). Springer.

Wang, Z., & Zlatanova, S. (2013b). Multi-agent infrastructure assisting navigation for first responders. In *Proceedings of the Sixth ACM SIGSPATIAL International Workshop on Computational Transportation Science* (pp. 1–6). ACM.

Wang, Z., & Zlatanova, S. (2013c). Taxonomy of navigation for first responders. In *Progress in Location-Based Services* (pp. 297–315). Springer.

Wang, Z., Zlatanova, S., Moreno, A., van Oosterom, P., & Toro, C. (2014). A data model for route planning in the case of forest fires. *Computers & Geosciences*, *68*, 1–10.

Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. *The knowledge engineering review*, *10*, 115–152.

Zeddini, B., Yassine, A., Temani, M., & Ghedira, K. (2008). An agent-oriented approach for the dynamic vehicle routing problem. In *International Workshop on Advanced Information Systems for Enterprises* (pp. 70–76). IEEE.