25

Modelling, Simulating and Autonomous Planning for Urban Search and Rescue

Vaccaro, J. and Guest, C.
University of California San Diego
United States of America

1. Introduction

This chapter presents a search and rescue simulation for a partially flooded city. The simulation makes use of novel approaches in the areas of modeling, simulation, and optimization. The model for the city is derived from Compact Terrain DataBase data. The section on modeling shows how this data can be preprocessed to make it suitable for simulation applications. Particular attention is given to preparing the data so that the resulting simulations can be run quickly and efficiently. The optimization section presents a hierarchical multi-agent planning and execution algorithm. The algorithm generates plans for multiple sorties of multiple agents. It then monitors the execution of each plan to provide on-the-fly improvements dictated by environmental uncertainties.

Parts of the flooded city remain above the water, but even some of these have become isolated because of flooded roads. City residents in these isolated portions must be rescued by boat or helicopter. In other portions of the city, one or more floors of buildings remain above the waterline, and the same options for rescue exist. In portions of the city that are above the waterline and remain connected to the mainland, there are still residents that require rescue. They may be too sick, injured, or frightened to evacuate themselves. These residents can be rescued with busses or helicopters. The simulation includes blockage of roads by high water and fallen trees at random locations. Helicopters can also be used for rapid reconnaissance to determine which portions of the city are accessible by road, and where to direct survey, supply, and rescue operations. They can also be used to deliver vital supplies rapidly to residents that cannot immediately be rescued.

2. Modelling

The environment modeled in this application is a small city that has been partially flooded. The city includes geological features such as rivers and frontage on a lake, as well as man-made features such as roads and buildings. All of this information is encoded in a general purpose format known as Compact Terrain DataBase (CTDB). CTDB is widely used for military and other applications.

CTDB represents three primary area types in urban terrain: natural terrain, developed terrain, and Multi-Elevation Structures (MES) (Witte, 2005). Natural terrain consists of soil type (modeled as triangles with vertices in three spatial coordinates), water (two-dimensional polygons), trees (points specified in two-dimensions), and canopies (two-dimensional polygons). Developed terrain includes roads, railroads, and canals, which are all defined as two-dimensional linear shapes. MES data includes buildings, monuments, and walls, which are defined as two-dimensional polygons with height. More detailed descriptions of the data format are given in the approach section.

For all CTDB data, urban terrain is gridded into square blocks (e.g., 256 by 256 meter squares) that divide many two-dimensional, and three-dimensional shapes into separate parts. This presents a problem for reconstructing the data into simulation objects, for example, treating buildings as a single object, and for keeping roads, rivers, and lakes properly connected. Since the simulation data is not intended for human use, but for computer processing, the visual aspects are less important than the object information. Creating simulation objects requires fusing fragmented shapes. Fusing shapes to create objects is done by drawing and filling them, which is described in detail below. Once objects are created, they form the basis for determining movement waypoints and connectivity.

The objective of preprocessing is to combine, segment, and repartition CTDB data into a compact format for simulation, while retaining the freedom of movement and connectivity of the original data. The freedom of movement is retained by creating waypoints at strategic locations, and connectivity is retained by connecting the waypoints in a manner that reflects their relationship in a traversable sense. For example, a road waypoint connected to a road waypoint signifies a traversable road, or a sky waypoint connected to a building top signifies a helicopter drop or rescue path. Using only waypoints and connectivity does not give the look and feel of a real life simulation for human use, but it does make all planning options available to computer algorithms.

The simulation uses agents of multiple types: busses, boats, and helicopters. For each type of agent, there is a corresponding set of waypoints. Agents of specific types are restricted to a particular set of waypoints, and paths. Planning of agent paths is simplified by clustering waypoints that are indistinguishably close together. This reduced set of waypoints is then preprocessed to determine shortest paths between all pairs of waypoints. Experiments have shown that reducing waypoint sets is important for efficiently computing the shortest paths for each waypoint type (Chandy & Mistra, 1982).

In this model, where each action choice has a pre-calculated shortest path, many options can be contemplated in successionto create a plan of action. This enables planning for many agents within an environment of multiple waypoint types.

This approach to modeling a high performance gaming environment to simulate S&R planning operations incorporates several desirable features: (1) a compact and computable data representation, (2) very fast simulation, (3) environment observation models, and (4) multi-agent planning. The following sections address each of these features in detail.

2.1 Feature One: Compact and Computable

A compact and computable environment representation is formulated through recomposing the CTDB into a network form, using waypoints and connections. CTDB data represents a virtual 3D world that human planners can navigate through with high visual acuity. For computer simulation purposes, there is no need to have a fully continuous and uniform representation; a tractable abstract representation is sufficient. Waypoints and connections provide a sufficient representation. Though the concept is simple, the challenge is developing a useful model that takes into account many different possible environmental features. For the example S&R application, only roads, rivers, buildings, open water and land areas, and terrain elevation are considered.

Roads and rivers are relatively simple to recompose, because they are given in CTDB as 2D linear segments with a width. Every curve requires many segments, while straight portions may be only one segment. Waypoints are created at both ends of each segment and are shared with all connecting segments. Connections between waypoints are simply the segments themselves. This approach works in most cases, but it is important to make sure that all segment intersections are included, because many times they are not represented explicitly. In such cases an additional waypoint is inserted at the intersection.

Rivers are a special case because they can be traveled by boat, but their banks can also be traveled by foot or may be flooded. Riverbank waypoints are generated in most cases by creating waypoints paralleling the river at a distance of half its width on both sides of the river. This does not work in the case of river intersections. These intersections are completed by adding a waypoint at the closest point between intersecting river segments that is beyond both rivers' widths.

Building waypoints and connections are implemented in three regions: interior, nearby exterior, and access points to the building. Given that many buildings are not single CTDB objects, but a combination of 3D volumes, merging is required. This is done by projecting all overlapping or connected volumes to the x-y plane (the ground) and then filling all internal pixels to determine the total footprint of the building object. Figure 1 illustrates this process.

To start, one building volume is chosen, and any other building sharing vertices with the chosen volume is selected. As they are added, selected buildings are eliminated from the list of unprocessed buildings. This is iterated until there are no more shared vertices. Next, the largest distance among all pairs of vertices of all the building volumes selected is computed, and a square image canvas with edge length twice that distance is generated to ensure that the object is filled correctly. The canvas grid is initially filled with all zeros. Next, each building volume is drawn and filled on the canvas with the value one. For example, filling each shape is performed by selecting the center-point and painting the object. If filling the object changes more than half the grid points on the canvas, then painting the location is outside the building object and undone and a new point to paint from is selected. Next, all interior building vertices are filtered out by keeping only those volume vertices that neighbor an outer pixel of value zero. Figure 1 shows the image of building volumes, where the single interior vertex is labeled as a 'triangle' and the exterior vertices are labeled as 'stars' and 'squares'. All 'triangle' coordinates are assumed as interior points and removed.

Remaining vertices are now reconnected such that they follow the outer wall. This is done by cycling through all connecting vertex pairs from all volumes, and keeping those pairs

that have both vertices on the exterior. For illustrative purposes, Figure 1 shows all the vertex pairs by placing a 'diamond' on either side of the center connections. Those walls with a diamond outside of the wall are kept. The heavier shaded walls in Figure 1 represent the kept walls.

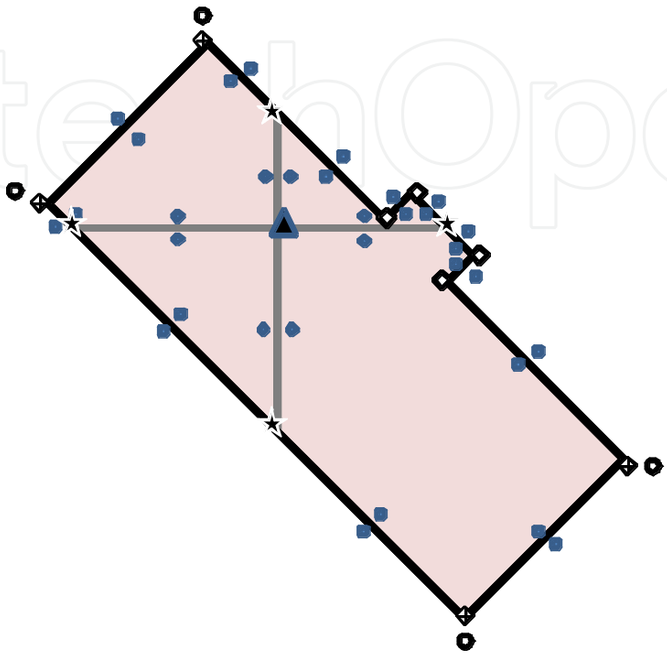


Fig. 1. Merging building volumes into one object

There are still some vertices that are in the middle of walls that can be eliminated without changing the outline of the building. Now that the complete perimeter of the building is known, the 'middle of wall' vertices can be eliminated by taking out all vertices where the walls on either side are colinear. 'Middle of wall' vertices are represented as 'stars' in Figure 1 and are eliminated.

There is now a building object with at least three connecting vertices, but often times more (i.e., in Figure 1 the 'hollow diamond' symbols represent a sufficient set of perimeter vertices). This process is continued from the first step until all building volumes are used.

For simplicity, only four exterior waypoints are defined near each building, unless only three vertices define the entire building object. These nearby waypoints are chosen by projecting the outermost perimeter vertices an additional two meters away from the building center.

The outermost vertices (corners) are chosen as follows: the first corner is the furthest from the center of mass of all filled points on the canvas, the second corner is furthest from the first corner selected, the third corner is furthest from the first and second corners in combined distance, and the fourth corner is farthest from the combined first, second and third corners. In Figure 1, the chosen corners are those with an 'x' in the 'square'. The nearby exterior building waypoints generated from these extreme corners are shown as 'circles' in

Figure 1. The connections among these waypoints are constructed such that they do not cross (i.e., minimal spanned distance). When buildings are very near one another, outside waypoints from multiple buildings are clustered if they are less than three meters apart. In other words, these nearby waypoints from multiple buildings are combined as one, while retaining their previous connectivity to other waypoints.

In CTDB, buildings often come with floor plans, but for this S&R application, we assume only two interior waypoints per floor, and two on the rooftop. One waypoint represents a location observable from the outside, while the other represents a hidden location. The pair of waypoints on each floor are connected. Connection between floors, exiting the building, and going to the roof are accessed only through the hidden waypoints. Both the hidden and observable waypoints are placed near the center of each floor, because observability is considered equal in all directions.

Fig. 2. City roads under maximum flood conditions

Building exit connections provide access to the nearest road for S&R evacuation purposes. These connections are found by considering each building exterior waypoint and determining the road nearest to it. Then the shortest of these lengths is selected as the road acces for that building. Buildings that share the same road waypoint are clustered together and considered as a single bus or boat stop. The stops are used in planning evacuations for each clustered set of buildings.

Open land and water area waypoints are necessary for regions where there are no roads or buildings. For simplicity, a hexagonal grid of waypoints with 75 meters edge length is projected on the 2D image of roads, water bodies, and buildings to provide a complimentary set of waypoints. All such waypoints within fifteen meters of an existing waypoint, on a road, in a water body, or within a building are removed. All other projected waypoints are kept as open area waypoints.

Connections between waypoints are implemented to complete the model. All waypoints, excluding building interior waypoints are connected using Delaunay triangles (Isenburg et al., 2006). Next, all Delaunay connections are projected onto the same 2D image used above. All connections that cross over roads, rivers or buildings are rejected and the remaining connections are kept. All connections discussed earlier (roads, rivers, buildings, etc.) are added to this connection model, and redundant connections are removed. This completes the surface model connectivity. Figure 2 illustrates a diagram of usable road connections during a flood at its highest level.

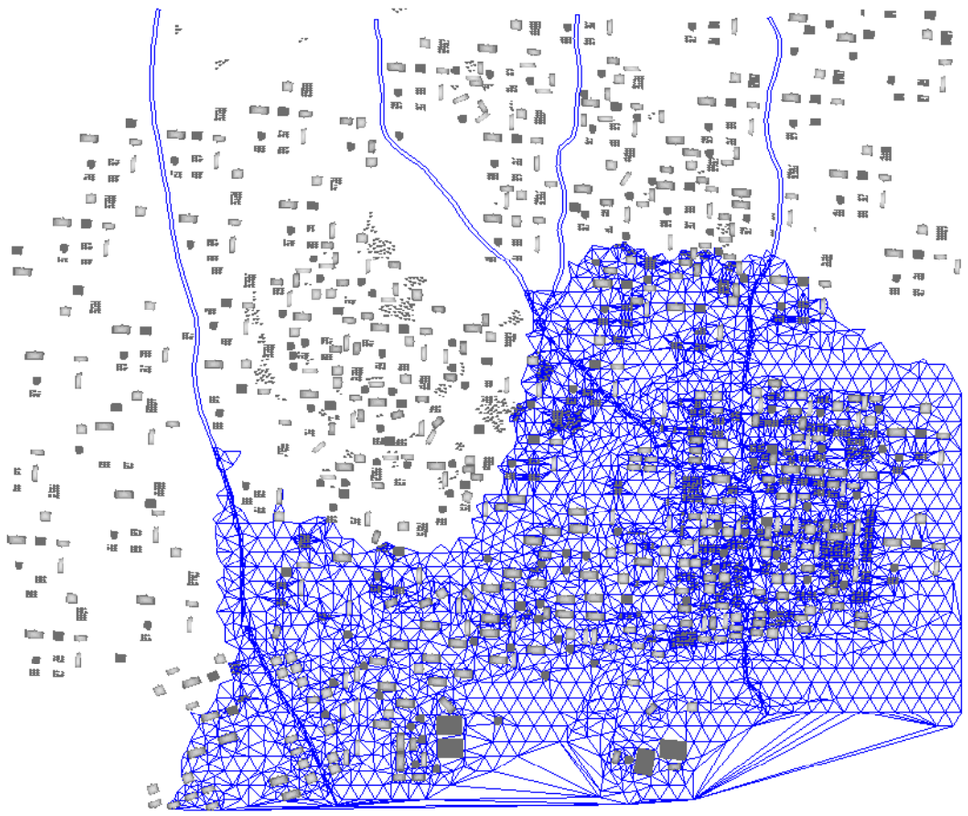


Fig. 3. City water routes under maximum flood conditions

Terrain elevation is also required in the model, because water level determines whether surface waypoints can be traveled by land or by water. In the example application, boats, busses and helicopters are used. Boats can traverse all waypoints below the waterline and busses can travel only road connections above waterline. Connections to and from buildings

are adapted to the lowest floor not underwater. Figure 3 illustrates the boat movement model during a flood at its highest level.

Helicopter paths require an air movement model. For simplicity, a sky waypoint is placed above each building at a constant elevation slightly above the tallest building height. Connections are made between each sky waypoint and to the building rooftop observable waypoint. In addition, a hexagonal grid of sky waypoints with 100 meters edge length is added wherever the added waypoint is not within fifteen meters of an existing sky waypoint. All sky waypoints are connected with one another via Delaunay triangles, but this time all connections are kept. A helicopter base is added at the most strategic (i.e., as deemed by an expert) open area waypoint and connected to the nearest sky waypoint. Figure 4 illustrates the helicopter movement model with all the sky-waypoints and connections, which allows any building to be visited.

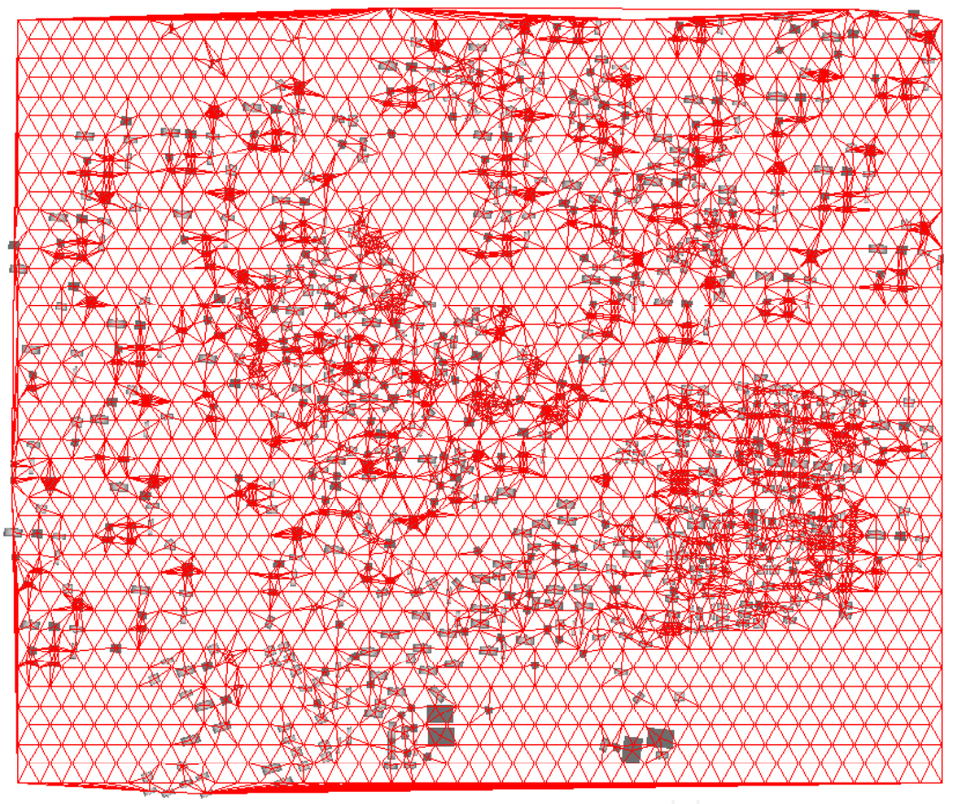


Fig. 4. City helicopter routes under any conditions

2.2 Feature Two: Very Fast Planning Simulation

Very fast planning simulation is enabled by pre-computing all shortest paths for all pairs of waypoints for boats, busses, and helicopters. Boat and bus paths can also be computed for different water levels, but for the example application only one water level was used. For simplicity, the positions of boat and bus stops are the same, whether they are above or below the waterline. Each vehicle also requires a base where rescued residents are dropped off, so boat bases are put at the highest point of each river, a bus base is placed at the highest

road waypoint. A helicopter base is placed at another strategic location. Figure 5 illustrates the bases and elevation terrain under normal conditions. The busses assume all routes are open as shown in this figure, but learn to adapt their planning based on the true available routes shown in Figure 2.

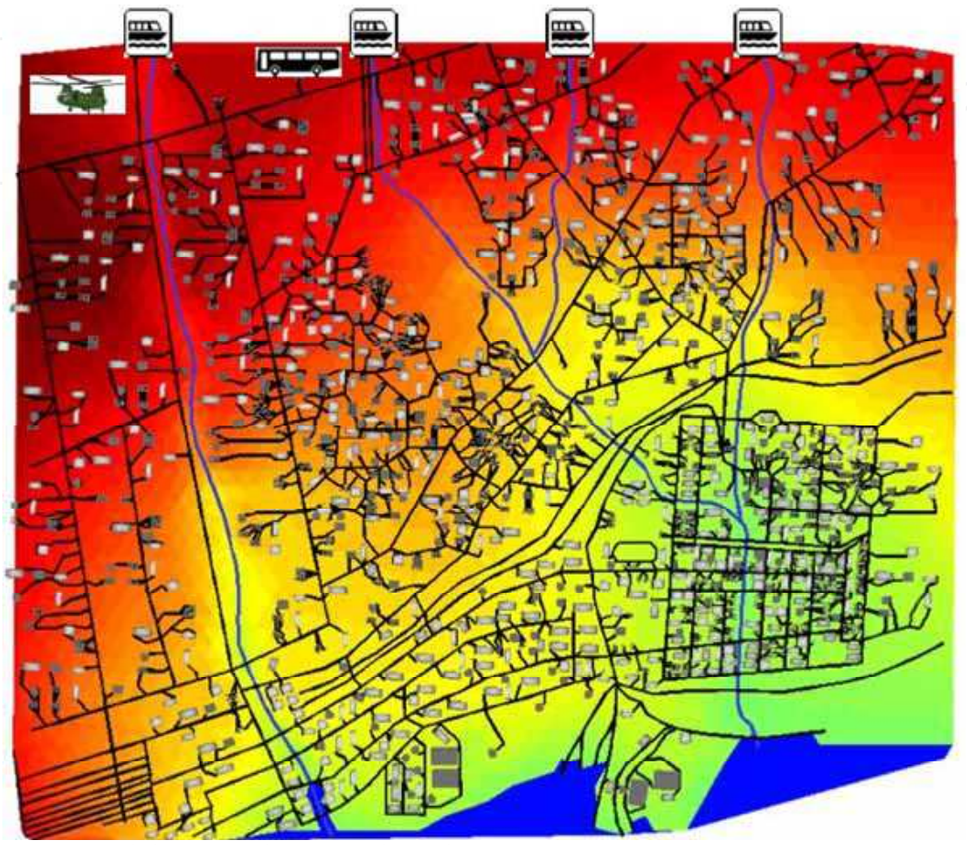


Fig. 5. City with bases under normal conditions

Preprocessing for shortest paths is done for all road waypoints for busses, all water waypoints for boats, and all sky waypoints for helicopters (Chandy & Mistra, 1982). This produces two matrices for each domain: a path matrix which indicates every waypoint one must traverse to go from any waypoint to another; and a distance matrix that indicates the distance between every pair of waypoints. The number of matrix lookups for a trip is the number of waypoints traversed on a path between the starting and ending waypoints. To begin, using the *path matrix*, cross reference the starting waypoint number as a row with the ending waypoint number as a column. This first lookup tells which waypoint to traverse first. In the same location in the *distance matrix*, the entry gives the distance traveled to the first waypoint. Next, lookup this new waypoint number in the path matrix as a row with the corresponding destination number as a column again, and iterate this procedure until the path matrix entry matches the destination.

In addition to preprocessing for the shortest paths, line-of-sight for helicopters is also preprocessed. Line-of-sight for boats and busses is not used, because they rely heavily on

sensing road access and building content by direct use of waypoints (can be considered as tactile sensing). Helicopters, on the other hand are not in direct contact with roads, water, or buildings, thus, they require a line-of-sight mapping. For simplicity, line-of-sight can only observe waypoints that are in close proximity (less than 50 meters on the surface plane), and is restricted as the speed of the helicopter increases. Flying over roads and water, a helicopter can determine whether the surface is available for boats or busses. Flying over buildings, a helicopter passing slowly enough, can determine that individuals are in the buildings, provided they are at the buildings' observable points. The probability of spotting people is inversely proportional to the helicopter speed. The probability of spotting someone within the 50 meter range is one minus three-halves the current speed divided by the top speed of a helicopter. The top speed of every helicopter is given as 200 kilometers per hour.

2.3 Feature Three: Environment Observation Models

Model observations include road conditions, waterway conditions, and the number of people in a building. Road observations determine which roads are open to bus travel. Water observations determine which waterways are open for boat travel. Buildings observations include people of four types: injured, unsupplied, supplied, and dead.

At the start of the simulation, there are 20,000 stranded people in buildings, and for simplicity, they do not move unless rescued. All people are randomly placed as follows: first, a floor location is chosen and one to ten individuals are placed in that location, with an average of 5.5. Both selection processes are random using a uniform distribution. This two step process continues until all 20,000 people are allocated.

At the start of the simulation, the people are divided into three categories: supplied, unsupplied, and injured. There are no dead people at the start of the simulation. The simulation assumes that half of the locations are supplied and half are not. Also, 5% of the locations have an injured individual. Thus, there are about 10,000 supplied and 10,000 unsupplied individuals on average, and about 180 injured individuals (20000/5.5 x 0.05). The times for survival of unsupplied and injured individuals are selected from the following probability density functions. For unsupplied, a natural lognormal distribution is used, with the mean (m) = 3.9 (exp(3.9) = 50 hours) and the standard deviation (sd) = 0.11, providing a range of 30 to 72 hours. For injured, a normal distribution is used, with the m = 16 (hours) and sd = 7, providing a range 1 to 36 hours. Values calculated as less than zero hours for injured individuals are reset to one hour, thus giving the injured at least some minimal lifespan. Injured and unsupplied people are on a time clock and die if not reached in time.

There are two sets of values for each observation type. One is the actual state, and the other is known state. The known state values are initialized to assumed values. The road model assumes all roads are open, and the water model assumes all low lying areas are flooded and traversable by boat. The building model assumes the person quantities are unknown for every floor of every building until observed. Using the same procedure described above, buildings not yet searched are randomly populated with people that are missing (those not accounted for from the 20,000).

Time Constraints /Delays	*Boats*	*Busses*	*Rescue Copters*	*Survey Copters*
Floor Search	300 sec	200 sec	600 sec	n/a
Building Evacuation	40 sec/ person	20 sec/ person	120 sec/ person	n/a
Road Blocks	n/a	600 sec	n/a	n/a
Backtrack	n/a	60 sec	n/a	n/a
Bad Plan	n/a	10 sec/ plan	n/a	n/a
Drop Off People	300 sec	300 sec	300 sec	300 sec
Drop Off Supplies	n/a	n/a	300 sec	300 sec
Turn Around	n/a	30 sec	n/a	n/a
Building Submerged	120 sec	n/a	120 sec	n/a
On Base No Orders	1200 sec	1200 sec	1200 sec	1200 sec
Off Base No Orders	120 sec	120 sec	120 sec	120 sec
Reinitialize Plan	600 sec	600 sec	600 sec	600 sec

Table 1. Time Costs and Delays

The known model learns real model information by tactile and visual sensing. Tactile sensing of travel routes is accomplished by boats and busses by visiting waypoints. As boats and busses follow connections, they discover whether they are open. Surface travel routes are also viewed by helicopters. Helicopters flying over surface waypoints can see if they are water or land, and slower helicopters can see more distant waypoints (range is 100 meters in the surface plane). For this application, survey helicopters travel much slower and have a better probability of seeing waypoint conditions below than do high-speed rescue helicopters. The probability of seeing roads is the same as spotting people as described above.

Agent or Vehicle Actuators	*Boats*	*Busses*	*Rescue Copters*	*Survey Helicopters*
Purpose	Search Bldgs; Rescue People	Search Bldgs; Rescue People	Search Bldgs; Rescue People; Drop Supply	Survey Road and Water Access; Spot People; Drop Supplies
Travel speed	40 km/hr	50 km/hr	100 km/hr	45 km/hr
Move	Water	Road	Sky	Sky
Range	4 hrs	8 hrs	1.5 hrs	1.5 hrs
Capacity	8 people	50 people	10 people	n/a
Supplies	n/a	n/a	1 Drop	12 Drops

Table 2. Agent or Vehicle Actuators

Tactile and visual sensing is also implemented for observing people. People are assumed to not travel without the aid of a rescue vehicle. Visual sensing is done only by helicopters. People can be spotted, but their type cannot be identified, such as injured, unsupplied, etc.

Tactile observations are done by searching buildings. Boats, busses, or rescue helicopters search a building on the first visit. They identify the occupancy of each building floor, and as many people as possible will be evacuated. Search times within buildings, and other time costs are shown in Table 1. Rescue capacities and other possible actions for each vehicle type are identified in Table 2. In summary, values with incomplete initial information include blocks in roads and water, and building contents.

2.4 Feature Four: Multi-Agent Planning

For S&R operations, multiple agents can travel by roads, waterways, and air to survey surface routes, spot, supply, or rescue people in buildings. Each agent has a purpose, a travel speed, a set of movement waypoints, a refueling range, a passenger capacity, and can possibly drop supplies. These parameter values are shown in Table 2. Each vehicle is an agent, and each agent type: boats, busses, and survey and rescue helicopters have different options, depending on their current state and observed models (building content and movement paths). More specifically, the options available depend on the accumulation of observations for movement and building content, and the current agent's state in regards to its location, fuel capacity, supplies available, and open passenger seating. Otherwise the vehicle is planned to return to base. A set of actions is selected as either a stop location or specific building, depending on the vehicle. Boats and busses must first locate a stop, and then choose to go to the building with that stop area. Helicopters choose buildings directly. A plan is formed when enough actions are chosen in succession for each agent until that agent must return to base. The total number of vehicles used in the simulation is arbitrarily set to twenty. Through the optimization process, planners learn how many of each vehicle type to requisition within the overall limit of twenty.

Available actions depend on the vehicle type and state of the vehicle. Boats and busses can choose any boat or bus stop as a destination, respectively. Boat and bus stops can serve multiple buildings. Plans for agents also need to select which buildings at each stop to visit. It is assumed that a boat or bus does not leave a stop until they are full of passengers, or all buildings at that stop have been searched. Helicopters are categorized as rescue or survey type. Rescue helicopters can choose any single building for search and rescue, and they drop a single supply package when they search the first building that has people. Survey helicopters survey roads, and spot or supply people within buildings. All vehicles return to base when their fuel time is spent, when the vehicle is full of passengers, or in the case of the survey helicopters, their supplies have run out, (see Table 2).

As agent actions are taken, the environment responds with time penalties, which are determined by travel speed shown in Table 2, and other time delays as shown in Table 1. Plans are made based on known observation values described in the previous section. Each vehicle's plan is considered complete when a return to base is included in the plan. Planning beyond one round trip for a vehicle has proven fruitless, because the real model values differ greatly from assumed values for long-term plans.

3. Example

There are four main features described above that went into implementing the S&R model. The best way to show the impact of considering these features is through example results.

Each feature is exemplified below by showing results that pertain to that feature. Results are based on playing out sixty games using a planner that picks random actions for each plan. If an agent plan encounters conditions that prevent its completion, a random, feasible, extension to the plan is provided. The planning section of this chapter addresses planners that learn to make effective, more focused choices.

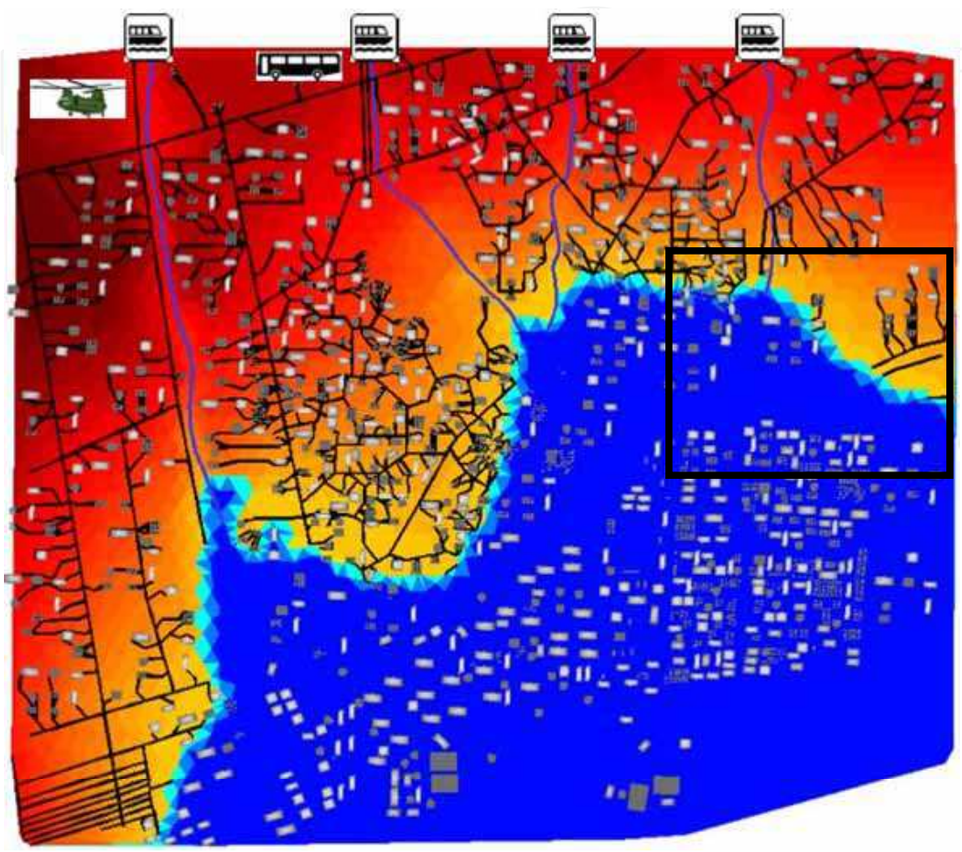


Fig. 6. City simulation example map

3.1 Feature One: Compact and Computable

To better understand the order of magnitude of this S&R problem, Figure 6 shows a top-down picture of the terrain model used. The city is approximately 4 km x 5 km. Normally, processing and computation is done over the terrain mapped in Figure 6, instead of the waypoints and connections derived from the CTDB data and illustrated in Figures 2, 3 and 4. Using waypoints and connections reduces the complexity of the space substantially by eliminating the waste of computing over every cubic meter of ground and space, and instead, computing action paths based on connected waypoint types. Within these 20 square km, there are 20,000 stranded people, twenty agents of various types, an initially unknown water level, road blocks, building locations where people require rescue, and bases to drop off people. Specifically, there are 3649 buildings with over 12,000 floor locations, 2135 bus stops, over 900 boat stops, four upstream boat bases, and a bus and helicopter base. There

are over 31,000 total waypoints for this application, instead of millions of cubic meters to compute on.

A cross-sectional view is shown in Figure 7. This view shows all waypoints and connections for a small grid section of the city, shown by the small square section outlined in Figure 6. Sky waypoints are shown on top and building connections, roads and waterways are shown below.

3.2 Feature Two: Very Fast Simulation

Due to the shortest path lookup tables described in Section 1.2, single actions for a plan are not waypoint to waypoint, but from origin to destination. This feature allows for very fast simulation speeds. The computation times are results from software developed in Matlab® particularly for this simulation, and are performed on a MacBook Pro running Matlab in a Microsoft Windows XP BootCamp environment. Complete S&R operations were run in an average of 7.66 minutes computer time, corresponding to 5525 minutes on average of real-world simulation time. That is a speed up of over a 700 times.

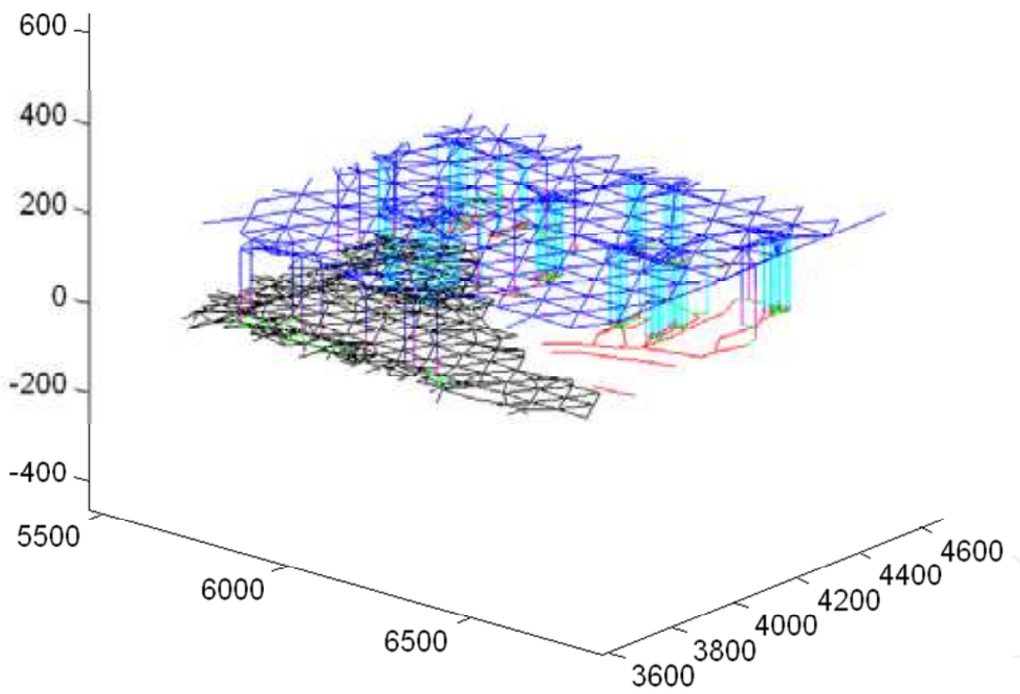


Fig. 7. Waypoint Connectivity

3.3 Feature Three: Environment Observation Models

Over the course of one example game, Figure 8 shows the changes in the observation models. The figure shows the accumulation of route information via both land and water, and the proportion of buildings searched and cleared, and people rescued over time.

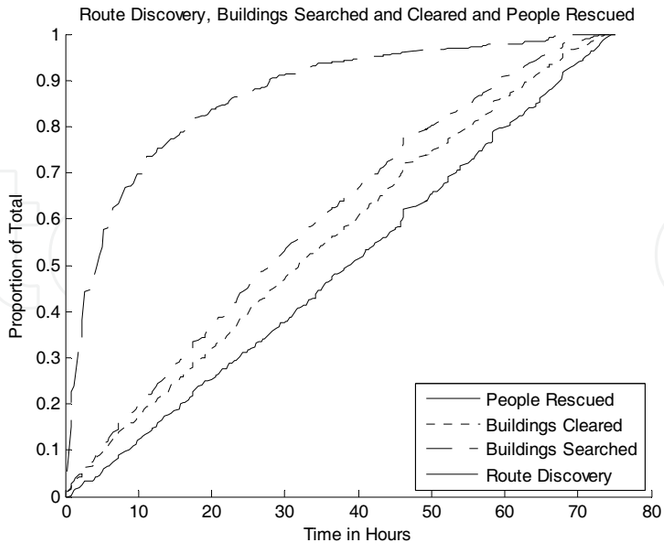


Fig. 8. Observation Models

3.4 Feature Four: Multi-Agent Planning

Over the course of sixty example games, there were twenty agents used in each game of various types. Over the course of an entire game, agents had plans of average length equal to about sixteen thousand visited waypoints. That is over nineteen million traversed waypoints for each of the twenty vehicles in sixty games.

4. Planning

The preceding sections of this chapter examined modeling a flooded city for a search and rescue operation. This section examines the development of an automated planner that directs the operation effectively and efficiently. Much as a human planner, the automated planner discussed here improves its performance through practicing many simulated search and rescue operations. To explore a variety of planning parameters, multiple automated planners compete against one another in tournaments. Characteristics shared by successful planners in one tournament are passed on to a new generation of planners in the next tournament.

4.1 Problem Domain

The search and rescue (S&R) problem is representative of a class of problems that has been difficult for automated planning systems. These problems share the following characteristics:

- Partially observable; the state of the environment is only partially known.
- Very large state-space representations ($> 10^{100}$)

- Finite but large set of available actions (> 10^3)
- Known state transitions (possibly stochastic)
- Finite set of measurable features
- A finite set of goals

Planning problems for large partially observable complex environments highlight three challenges in particular: (1) balancing exploration of the environment for new information against exploitation of obtained information to directly achieve goals; (2) determining where to assign credit in choosing actions, especially for long action sequences; and (3) providing a scalable framework, where the speed of the simulation is minimally affected by the number of agents or actions available.

In a partially observable environment, balancing exploration and exploitation is very important to the success of a complex mission. Several approaches have been tried. Reinforcement learning systems use temporal difference to consider action strings of length greater than one, but fail to deliver on plans that take a large number of steps and where the state-action pairs exceed 10^{20} combinations (Levinson et al., 1991), (Tesauro, 2002), (LaValle, 2006). Learning classifier systems focus on longer string plans, but usually use discrete rule sets and have not been tested on problems of the size presented here (over 10^{100} states) (Velagic, 2006). In the application presented, planning lengths are long, dozens of actions per iteration, and thousands over the course of one complete simulation. Also, uncertainty in the initial state is large, thus the search for a better planner requires generalization over the state space instead of formulating specific state-action pair rewards.

The credit assignment problem is also important in planning for large partially observable environments. Planning has three phases: generating, executing, and evaluating plans. How to merge these three aspects into the planning cycle will be described in detail. The form of the solution is important in answering many questions related to assigning credit: How often should the planner re-plan? How often should the model be updated before re-planning? How will the different types of agents cooperate? What features of the environment are important factors in developing a good plan?

4.2 Approach

The approach for developing an effective automated planner has five tiers, from the inner cycle of dynamic planning, executing, and evaluating plans for planners and agents, through the highest level of adapting planners' strategies using tournament play of multiple games. Figure 9 illustrates this ADP&E implementation framework.

The core cycle (1) is concerned with agent action and environment response., Individual agent action sequences are planned, executed, and evaluated in the model environment, over many cycles, and for all agents in the correct time sequence. At the second level, agents execute a given plan. Third, the planner is the conceiver and conductor of a plan. A planner has a set of parameters that determine its choice of planned actions, and how often to re-plan those actions. Fourth, a game is a theatre where action sequences can be executed in the real model environment. The final goal state must be achievable, because human

intervention is prohibited in this framework and a game only completes when the final goal is achieved. Fifth, tournaments of games are arranged, so that planner parameter settings can improve over the course of many tournaments. Through evaluating each planner's progress, and modifying the best planner's parameters, planners can improve their effectiveness.

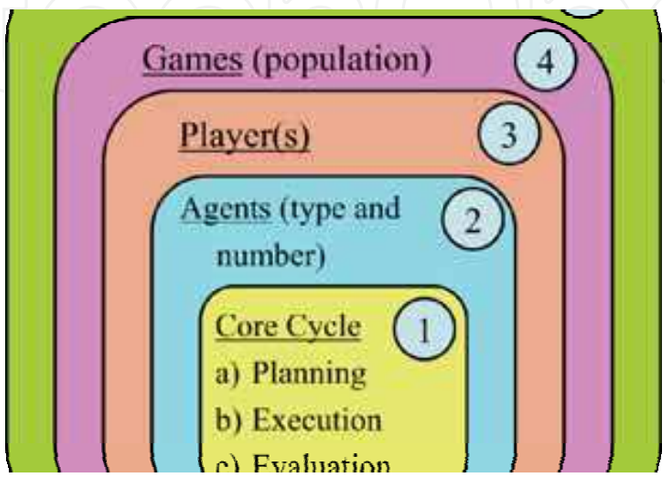


Fig. 9. ADP&E Implementation Framework

All five tiers are interdependent in the time domain. In a typical computing environment, the core cycle builds plans and performs action-response simulations within milliseconds of computation time. Agent operations of many planned actions are performed within seconds. A planner completes many necessary operations to an endgame within minutes. A game set for one tournament generation takes hours. A completed tournament, where planners' parameters converge on a successful strategy takes several days. The computation times are results from software developed in Matlab® for this paper and are provided to indicate relative scales.

4.2.1 Tier One: Core Cycle

At the heart of this approach is the core planning cycle. Figure 10 illustrates this cycle. The core cycle has three components: (1) *plan-generator* (PG); (2) *plan-executor* (PX); and (3) *plan-evaluator* (PV). The plan-generator strings together individual actions to form plans for all agents. The plan-executor executes the planned actions in time order for a period that completes a single task for one of the agents. A task is considered as a search and rescue operation for a single building.. The plan-evaluator evaluates how well the remaining plan will execute given new information acquired as a result of the partially executed plan.

The three core-cycle components share three internal objects. These three objects are *plans*, *models*, and *expectations*. Plans are generated by PG, executed by PX and evaluated by PV

repeatedly until a simulation is complete. Models are used in PG to predict future states, are used in PE to observe the real environment states, and are used in PV to observe whether expectations have been met. The models used in PG and PV are virtual-state models, which initially contain only part of the information in the real-state model used in PX. The real-state model is where a plan is executed. Virtual-state models do not include many real state values until observed. Expectations measure how well a plan achieves a desired goal. Expectations are predicted in PG and compared to actual progress in PV. Each agent has an expectation for its plan. If, during execution, expectations are met to a prescribed degree, a plan is retained; otherwise a plan is reformulated in PG. The motivation to retain a plan is a tradeoff of planning-time verses the risk associated with continuing the current plan. This tradeoff parameter is called risk aversion and is part of the planner that will be discussed in detail later.

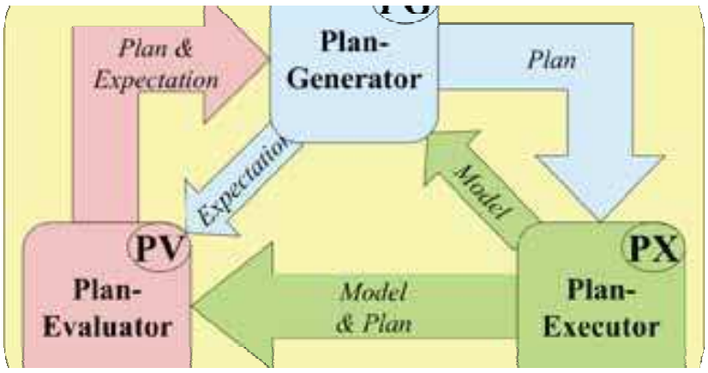


Fig. 10. Planning Core Cycle

4.2.2 Tier Two: Agents

Agents are the instruments of the plans. Agents possess various actuators and sensors. For the S&R problem, the agents are the vehicles used in the model environment. Tables 2 and 3 illustrate all the agent categories and their associated characteristics. Table 1 shows the agents' temporal parameters. Each agent has an individual plan, a timestamp, a location, a fuel level, a passenger list, a speed, and a remaining time constraint.

4.2.3 Tier Three: Planners

The planner has a set of control parameters that affect generating, executing, and evaluating plans. Each action within a plan is chosen according to the planner's characteristics. The planner's characteristics are defined by parameters used in four areas: (1) model update time, (2) risk aversion for re-planning, (3) numbers of each agent type given limited resources, and (4) the nine decision features described in this section.

For partially-observable environments, the model update parameter is paramount. It determines how frequently the virtual-state model is updated to more closely resemble the true real-state model of the environment. The information in these updates comes from the sensor input of the agents. This refresh enables the planner better to predict results of future

actions. For this application, the model update parameter was not adapted but set to a relatively small time increment. Specifically, every time a vehicle returned to base, all agents were required to report all of their newly acquired sensor data (about 20 minutes real-world time). This assumption was necessary, because results showed that not updating the virtual model frequently enough caused worse results than randomly searching with continual updates.

Another important planner parameter is risk aversion, r, (0 <= r <= 1). Risk aversion trades time required to plan against the desire to re-plan. This competition is based on whether a plan is meeting expectations as evaluated in PV. When the risk aversion parameter is high, the tendency is to re-plan often, when risk aversion is low, the tendency is to stick to the current plan and thereby avoid the time for re-planning. Specifically, if the risk aversion parameter is set to one, then expectations must be met completely or the planner will re-plan. If the risk aversion parameter is set to zero, then the planner will not re-plan unless the agent no longer has a plan.

The numbers of each agent type to use in a partially observable problem is unknown and therefore made adaptable. Experiments have shown that an unrestricted number of agents leads to minimal loss of life and therefore trivializes the problem. Therefore, as in the real-world, resources are limited, and in this particular case, only twenty agents of all types are allowed.

Observable factors in the execution environment that are relevant to the planning process are called features. The nine decision features to choose a destination are: (1) action distance, (2) nearest base distance, (3) nearest same type vehicle distance, (4) people present, (5) injured present, (6) presently unsupplied, (7) people spotted, (8) preceived survival time, and (9) previously visited by boat or bus. These features were used because they are quickly calculable from the environment for each possible action and seem relevant to choosing actions. At each time step an agent has thousands of potential actions (3649 buildings) to choose from, and plans may have action sequence lengths in the dozens. This makes a tree search method impractical. Nor does one want to restrict an agent to follow a deterministic rule set based on any specific feature or combination of features, because that would severely limit the search space of all possible plans. Therefore, a new method was developed that does not restrict the search space. It combines a Monte Carlo (MC) method of choosing actions and weighted Beta distributions (WBDs) (Vaccaro & Guest, 2008) for representing decision preferences.

For our purposes, MC represents taking a random choice from many possibilities, and the WBDs represent the density functions of actions from which to make those choices. Each WBD required for each feature and agent type has three parameters: w, α, and β. The probability mass function of the WBD is as follows:

$$f\left(x; w, \alpha, \beta\right) = w\frac{\Gamma\left(\alpha+\beta\right)}{\Gamma\left(\alpha\right)\Gamma\left(\beta\right)}x\left(1-x\right)^{\beta-1} \tag{1}$$

where α is the alpha parameter, β is the beta parameter, w is the weight parameter, x is a vector of feature values corresponding to all possible actions, and Γ is the Gamma function (Nadarajah & Kotz, 2007).

MC implements taking a random choice from many possibilities and the WBDs represent the probability density functions of choosing particular actions from among those choices. The WBD for each feature and agent type has three required parameters. Boats and busses use only the first eight of the nine decision features listed above, while both helicopter types use all nine features. Thus, there are 34 WBDs required, and thus, 102 WBD parameters for this S&R problem. The three WBD parameters are weight, alpha and beta; all initial values are randomly chosen. For each agent, each weight is greater than or equal to zero and all weights sum to one. The alpha and beta parameters are restricted to be greater than zero and give a variety of possible distributions, as shown in the results section. The reason WBDs are used is that they do not restrict the search space if the initial parameters are correctly set. For instance, if all WBDs have alpha and beta parameters set to {1, 1}, then the probability of selecting any action is equally likely at every time step, independent of the weightings.

The process of using MC and WBD together is as follows. First, for each agent under consideration, a mass function is calculated for each feature type for all possible actions (equation above without weight w). The mass function is a vector of length equal to the number of possible actions and each mass represents its particular influence in action selection. Before w can be applied, each mass function is normalized to sum to one, to represent a probability mass function. Next, each mass function is multiplied by the corresponding weight (w) assigned to that agent and feature. Finally, all resulting mass functions corresponding to each agent are summed. These new mass functions, which holds the accumulated influence of all features for each agent, is normalized and expressed as a distributed mass function (0, 1) to form a reference lookup table for each action. At this point the selection process begins. First, a random number is chosen from a uniform density (0, 1), and this random variable chooses the action by looking up the corresponding action linked to that number in the distributed mass function table. Note that, the greater a feature's weight, the greater its influence, and the BD settings focus on influential feature values.

In summary, a planner has a risk aversion parameter and four parameters that represent the numbers of each vehicle type. Since the boat and bus agents use eight of the nine decision features, there are 24 WBD parameters (8 weights, and 16 alpha and beta) for both boat and bus type vehicles. Since helicopters use all nine features, there are 27 WBD parameters (9 weights, and 18 alpha and beta) for both survey and rescue helicopter vehicles. Thus, the total parameters for a planner number 107 = 1 + 4 + 24 + 24 + 27 + 27.

4.2.4 Tier Four: Games

The fourth tier in this implementation framework is games. For this S&R application, a game is the complete evacuation of the city. The game level is where results are accumulated to determine the effectiveness of a planner. Results include the planner parameters described above, and how many residents died in the S&R operation.

4.2.5 Tier Five: Tournaments

The fifth tier is tournament play. The tournament tier compares the accumulated results from all planners in a generation. For the S&R application, each generation is a sixty game tournament with twenty planners competing. Each simulation run uses a different

randomly initialized environment. Each planner plays three games to determine its effectiveness in generalizing for partially observable environments. The score for each planner is the maximum number of dead residents for all three games. Obviously, a low score is better. Improving the planners' capabilities is adapted using a genetic algorithm. The genetic algorithm follows these rules:

The top 4 performing planners return for next generation

Next 4, mutate decision feature weights {34 parameters in all}, vehicle number {4} and risk aversion {1}

Next 4, mutate the BD parameters {68 parameters}

Next 4, crossover decision feature weights {34 parameters}, vehicle number {4} and risk {1}

Next 4, crossover the BD parameters {68 parameters}

The weights, risk aversion, and vehicle number are mutated differently. Each weight is adjusted by adding a random variable selected from a normal distribution with mean zero and standard deviation 0.2. The weight is adjusted not to be negative or exceed one. The weights are then renormalized to sum to one.

Each risk aversion parameter has a 50% probability of being changed. If selected to change, it is adjusted using a normal distribution with mean zero and standard deviation 0.05; not to be negative or exceed one. Each number of vehicle type parameter is also adjusted with a 50% probability. Each vehicle type has the probability of incrementing or decrementing the number by one, provided the number is greater than zero and the numbers of all vehicle types sum to twenty.

Mutation of the BD parameters is a special case, because the values must only be greater than zero. Thus, a log normal density function is used to pick the adjustment in the alpha and beta parameters. The mean of the density function chosen is 2 over the generation number and the standard deviation is the square root of the mean. This lognormal density produces an adjustment in the BD parameters to be very large at first and decrease with each subsequent generation. Experimental results have shown this to be prudent in providing enough variability in the BD densities.

For all crossovers, a planner is picked based on how well it did with respect to other planners. Planners with better than average performance are put in a pool with probability of selection proportional to their score above the average. Each selected planner is combined with each top planner using crossover. The new planners are initially a copy of each selected top planner. Crossover is performed by cycling through each parameter and giving a 50% chance of acquiring a parameter from the randomly chosen planner paired with the top planner.

The genetic algorithm runs for many tournament generations until the planners' parameters show some degree of convergence, or when the capability of the planners seems to show little progress in improvement.

5. Simulation Example

The process of developing an effective planner is better understood by describing the example implemented for this application. Note that in Table 1 helicopters hovering over buildings take far longer to search buildings than busses or boats, and busses are by far the quickest. Busses evacuate the buildings the quickest per person, because they do not have to deal with water or hovering in midair. Busses are the only vehicle that experience problems with road blocks and have additional time penalties associated with making plans to travel on unknown roads. Only helicopters drop off supplies.

Agent or Vehicle Sensors	*Boats*	*Busses*	*Rescue Helicopters*	*Survey Helicopters*
Route Access via Trial	YES	YES	n/a	n/a
Route Access via Sight	NO	NO	LIMITED	YES
Survivors via Search	YES	YES	YES	NO
Survivors via Search	NO	NO	LIMITED	YES

Table 3. Agent or Vehicle Sensors

Decision features one, two and three are measurements of distance. These decision features have a continuous range of values (i.e., zero to 6500 meters). Note that "Action Distance" is for a particular leg of the trip, and "Nearest Vehicle Distance" pertains only to same type vehicles.

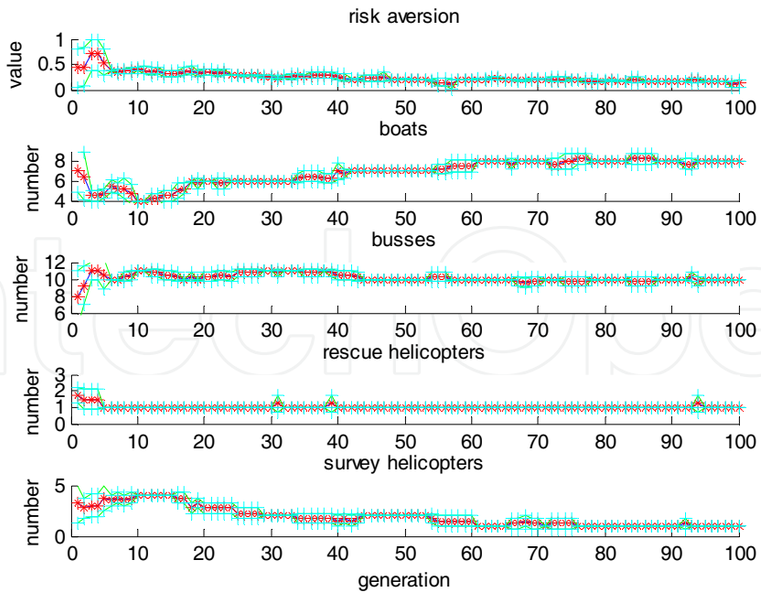


Fig. 11. Risk Aversion and Number of Vehicle Types

Since many stops and buildings are initially unsearched and have varying numbers of people left behind in a building once searched, decision features four, five, and six are multi-valued. For unsearched buildings, a value of negative one is used and for searched buildings the range of values used is distributed as $\log_{10}(PLB+1)$, where PLB is the number of people left behind. For feature seven, all stops and building locations are assigned zero unless people are spotted. For instance, if a helicopter spots a person at a building, then the value assigned to that building is one. However, for stops it is different. For a stop, the fraction of buildings in which people have been spotted at that stop is used instead. For feature eight, survival time is the projected amount of time available for unsupplied or injured people before they die. An estimated time is given for all stops and buildings, then, as people are discovered, their projected lifespan time is used to update the feature. It is presumed that all unsupplied and injured people will die within 75 hours. Thus 75 hours is assumed for all unsearched locations. This feature value will change with time. People not retrieved in time will have a negative lifespan and will be dead if their building was previously searched, and presumed dead for an unsearched building. Buildings where supplies are dropped add 48 hours to the life span for unsupplied people in those buildings. Injured people must be rescued to survive, because supplies do not extend their lifetime.

Feature nine, is a simple binary value. If a boat or bus has searched a building, then the value is one, otherwise the feature value is zero.

All feature values are normalized to be between zero and one. For decision features one, two and three, the largest diagonal distance across the city is used to normalize the decision features (6500 meters). All other feature values are normalized by subtracting the minimum and dividing by the range. For example, for feature eight, all values are kept between -100 and 100, thus, 100 is added and the result is divided by 200.

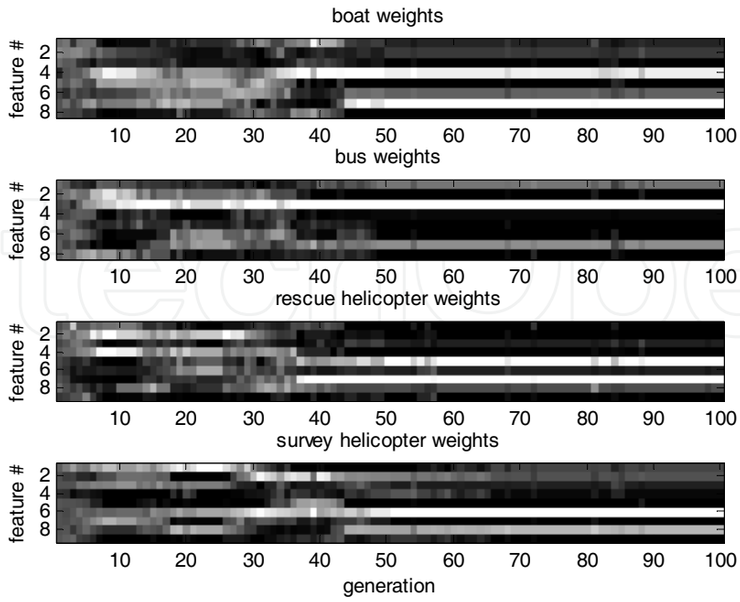


Fig. 12. Decision Feature Weights

The number of vehicles of each type is not predetermined, but the total is limited to twenty vehicles, with a minimum of one vehicle of each type to ensure that the game will run to completion. A game run completes with the evacuation of all buildings.

6. Results

The results are in five areas: (1) risk aversion, (2) number of vehicle types, (3) decision feature weights, (4) decision feature beta density functions, and (5) resulting evacuation progress in saving lives. All results are for 40 generations of tournament play.

The risk aversion parameter is used to trade off time to plan against failure of the current plan to meet expectations. Time to plan is fixed at ten minutes of simulated time per plan per agent. Expectations are based on a point system where an agent is given three points per injured person rescued, two points per unsupplied person, one point per supplied person, and zero points per dead person rescued. The evolution of the risk aversion parameter is shown in the top graph of Figure 11. The final converged value is around 0.25, thus, if a plan achieved just 25% of its expected points, replanning was not triggered. In another experiment, a 20 minute penalty for replanning produced a zero risk aversion level, meaning the time to plan was too costly to ever re-plan unless there was no way to proceed with the current plan.

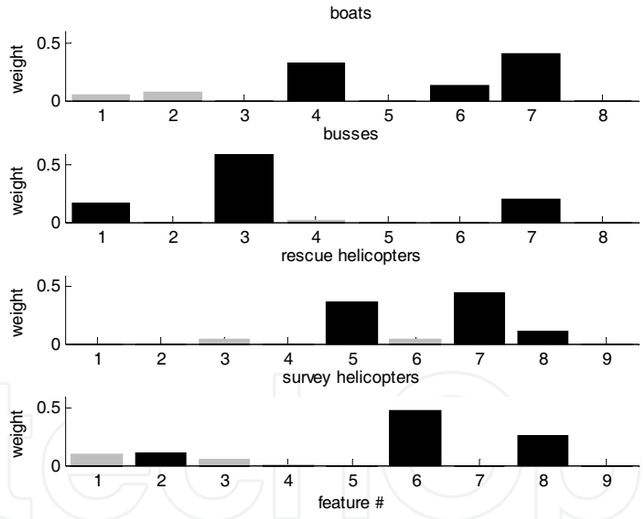


Fig. 13. Final Decision Feature Weights

The numbers of vehicle type are shown in the four lower graphs of Figure 11. In later generations, boats number around seven, busses around ten, rescue helicopters around one, and survey helicopters around two.

The decision feature weights are shown in Figures 12 and 13. Figure 12 presents the change in weights over tournament generations. Note that in all image figures (12, 14, 15, 16, and 17), that lighter shades indicate higher values. Figure 13 illustrates the final weights observed in generation forty. The three black bars indicate the three largest weights. Also,

that the feature numbers on the left of Figure 12, and on the bottom of Figure 13 are the same as displayed in Figure 11.

The decision feature beta density functions shown in Figures 14, 15, 16, 17 and 18 correspond to the black bars in Figure 6. This points out the decision features that exhibited the highest influence. Figures 14 through 17 shows the density functions across all forty generations, while Figure 18 shows the final density functions. Note that sometimes the shading goes dark in Figures 14 through 17. This indicates that the weight was low for that generation (Figure 12) and had little influence. Figure 18 provides a good summary of influential features. Each graph directly corresponds to a black bar in Figure 13.

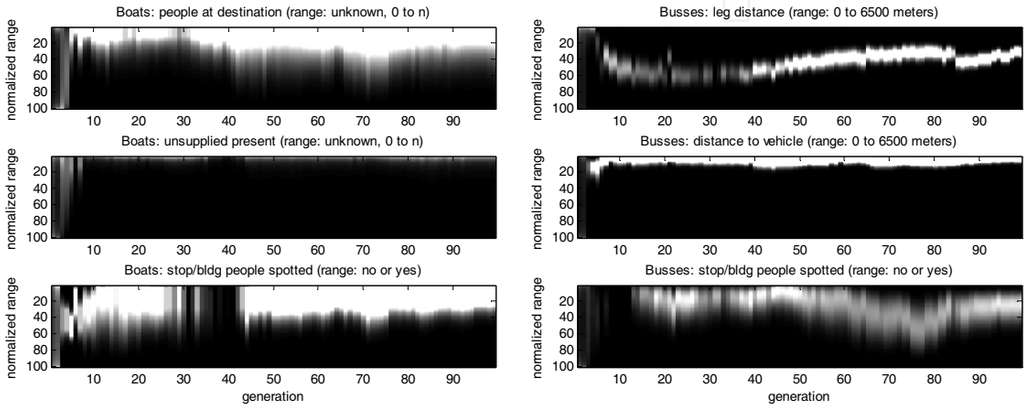


Fig. 14 and 15. Boat Features Weighted Beta Densities and Bus Features Weighted Beta Densities

The feature densities for boats are in the top row of graphs of Figure 18. The first graph indicates that since boats are slow, they tend to take tasks nearby. The second graph indicates that boats are also faster at searching buildings than helicopters, so they are sent to buildings that need searching. The third graph indicates that it is better for a boat to go to a building that has enough supply time, and therefore fewer dead.

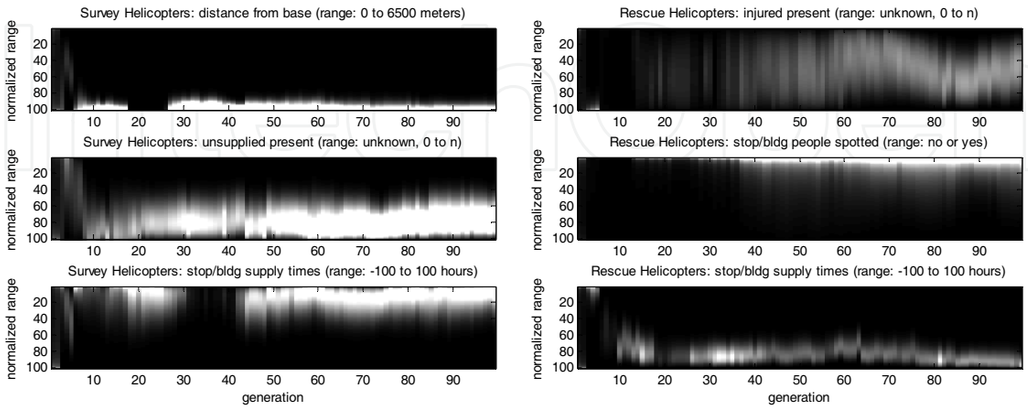


Fig. 16 and 17. Rescue Copter Features Weighted Beta Densities and Survey Copter Features Weighted Beta Densities

The feature densities for busses are in the second row of graphs in Figure 18. The first graph strongly indicates a need to choose actions near other busses. Since busses encounter road blocks due to trees and water, staying near other busses increases the chances of finding an open path. The second and third graphs in row two indicate a slight additional preference to go where few people are injured and where people have not been spotted.

The feature densities for rescue helicopters are in the third row of graphs in Figure 18. The three graphs indicate a preference to go to buildings where people are injured, where no one has been spotted, and where people still have supply time remaining.

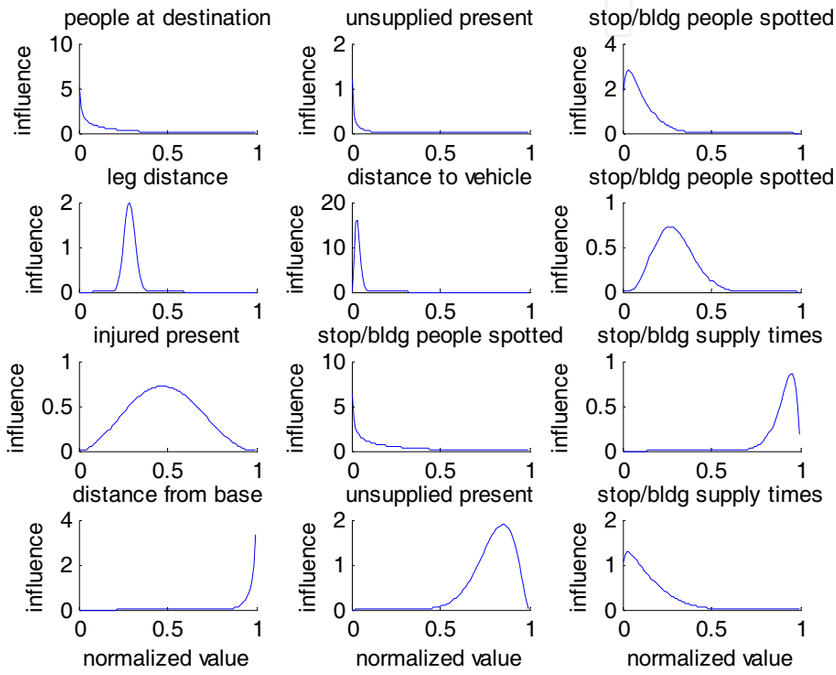


Fig. 18. Largest Feature Weighted Beta Densities

The feature densities for survey helicopters are in the final row of graphs of Figure 18. The three graphs indicate a preference to go far from the base, to go where people are not injured, and to go where people require supplies. Since the busses cannot go far from the base due to water levels, it makes a lot of sense for helicopters to supply people far from the base first.

The evacuation progress of the top performering planners is shown in Figure 19. Evacuation progress improves drastically over the first few generations and then slowly after that.

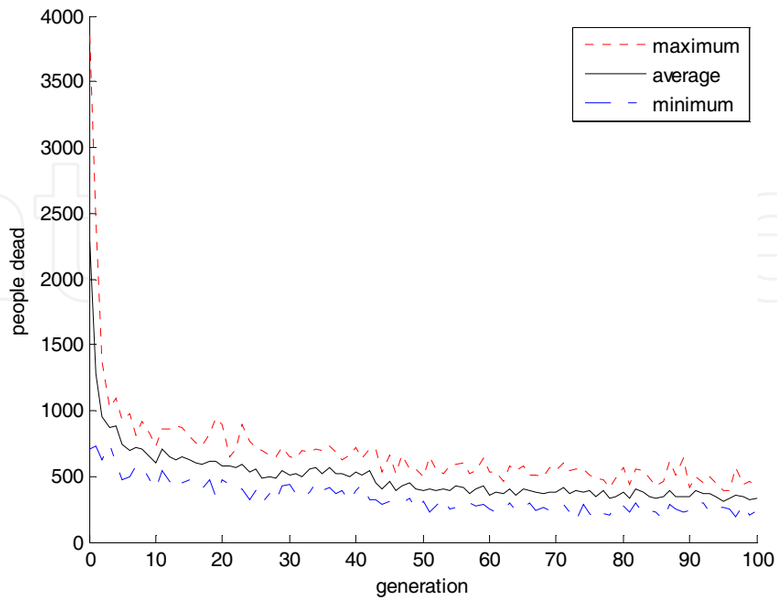


Fig. 19. Evacuation Progress of Top Four Performers

7. Conclusions

The chapter presented a three main advances: (1) a modeling approach for generating an urban terrain model from a Compact Terrain DataBase (CTDB) for computer-simulation of a search and rescue (S&R) operation, (2) a implementation strategy for integrating the model into an autonomous dynamic planning and execution (ADP&E) framework for gaming simulations, and (3) an evolutionary strategy for using simulation results to improve the ability of the planner. This approach is not considered an optimal strategy given that the approach has only been applied to a partially observable S&R operation described here. Many advances can be made to improve the generalization of this approach to more problems with multiple planners.

8. References

Witte, T.; *A survey of 3-D urban mapping and visualization capabilities from an army perspective*, Proceedings of the ISPRS Joint Conference, V. XXXVI, March, 2005.

Chandy, K. M., Mistra J.; *Distributed computation on graphs: shortest path algorithms,* Communications of the ACM, V. 25, No. 11, November, 1982, pp. 833-837.

Isenburg, M., Liu, Y., Shewchuk, J., & Snoeyink, J.; *Streaming computation of Delaunay triangulations*, ACM Transactions on Graphics 25(3), July 2006, pp. 1049-1056.

Levinson, R., Hsu, F. H., Schaeffe, J., Marsland, T. A., & Wilkins, D. E.; *The role of chess in artificial-intelligence research*, ICCA Journal, V. 14, N. 3, 1991, pp. 153-161.

Tesauro, G.; *Programming backgammon using self-teaching neural nets*, Artificial Intelligence, V. 134, 2002, pp. 181-199.

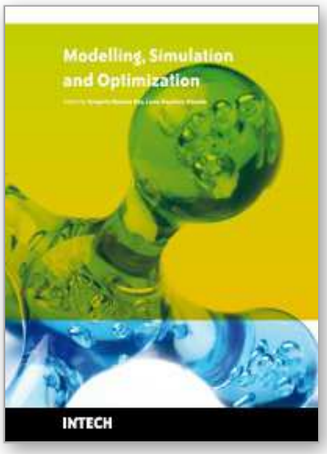
LaValle, S. M.; *Planning Algorithms*, Book: Cambridge University Press, 2006.

Velagic, J., Lacevic, B., Osmic, N.; *Efficient Path Planning Algorithm for Mobile Robot Navigation with a Local Minima Problem Solving*, International Conference on Industrial Technology, December, 2006, pp. 2325-2330.

Vaccaro, J., Guest, C.; *Automated Dynamic Planning and Execution for a Partially Observable Game Model: Tsunami City Search and Rescue*, World Congress on Computational Intelligence (WCCI'08), June 2008.

Nadarajah, S., Kotz, S.; *Multitude of beta distributions with applications*, Statistics: A Journal of Theoretical and Applied Statistics, Volume 41, Issue 2 April 2007, Pages 153-179.

IntechOpen



Modelling Simulation and Optimization

Edited by Gregorio Romero Rey and Luisa Martinez Muneta

ISBN 978-953-307-048-3
Hard cover, 708 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

Computer-Aided Design and system analysis aim to find mathematical models that allow emulating the behaviour of components and facilities. The high competitiveness in industry, the little time available for product development and the high cost in terms of time and money of producing the initial prototypes means that the computer-aided design and analysis of products are taking on major importance. On the other hand, in most areas of engineering the components of a system are interconnected and belong to different domains of physics (mechanics, electrics, hydraulics, thermal...). When developing a complete multidisciplinary system, it needs to integrate a design procedure to ensure that it will be successfully achieved. Engineering systems require an analysis of their dynamic behaviour (evolution over time or path of their different variables). The purpose of modelling and simulating dynamic systems is to generate a set of algebraic and differential equations or a mathematical model. In order to perform rapid product optimisation iterations, the models must be formulated and evaluated in the most efficient way. Automated environments contribute to this. One of the pioneers of simulation technology in medicine defines simulation as a technique, not a technology, that replaces real experiences with guided experiences reproducing important aspects of the real world in a fully interactive fashion [iii]. In the following chapters the reader will be introduced to the world of simulation in topics of current interest such as medicine, military purposes and their use in industry for diverse applications that range from the use of networks to combining thermal, chemical or electrical aspects, among others. We hope that after reading the different sections of this book we will have succeeded in bringing across what the scientific community is doing in the field of simulation and that it will be to your interest and liking. Lastly, we would like to thank all the authors for their excellent contributions in the different areas of simulation.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China

51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen

IntechOpen