A Novel Search Strategy for Autonomous Search and Rescue Robots

Sanem Sarıel[1] and H. Levent Akın[2]

[1] Istanbul Technical University, Dept. of Computer Engineering, Istanbul, Turkey
sariel@cs.itu.edu.tr
[2] Boğaziçi University, Dept. of Computer Engineering, Istanbul, Turkey
akin@boun.edu.tr

Abstract. In this work, a novel search strategy for autonomous search and rescue robots, that is highly suitable for the environments when the aid of human rescuers or search dogs is completely impossible, is proposed. The work area for a robot running this planning strategy can be small voids or possibly dangerous environments. The main goal of the proposed planning strategy is to find victims under very tight time constraints. The exploration strategy is designed to improve the success of the main goal of the robot using specialized sensors when available. The secondary goals of the strategy are avoiding obstacles for preventing further collapses, avoiding cycles in the search, and handling errors. The conducted experiments show that the proposed strategies are complete and promising for the main goal of a SR robot. The number of steps to find the reachable victims is considerably smaller than that of the greedy mapping method.

1 Introduction

The disasters such as earthquakes make clear the necessity of having robust, dynamic, and intelligent planning systems and powerful human-machine interaction. In general, the scale of the disaster and the speed of changing situations are far beyond the capabilities of human-based mission planning [3]. Often the lack of qualified rescue workers is a major problem. In addition, there are many difficulties for human team members in disaster areas such as dangerous or unreachable places. Although rescue dogs could help reduce the human risk by searching smaller voids in the rubble than a human can, in many cases a video camera or any structural-assessment equipment is more useful [6]. In overcoming the difficulties mentioned above, robots can be very sutiable [2].

The research on this area has become very attractive during the last decade. RoboCup-Rescue League has been started in 2001 [7]. Many search and rescue (SR) robot architectures, the variety of which is very large [3], have been proposed. In these proposals, the focus is typically on mechanical design, and the planning strategies are not addressed in detail.

As the most important part of a SR robot, the planning layer should be capable of generating effective plans for finding victims in a short period of time

D. Nardi et al. (Eds.): RoboCup 2004, LNAI 3276, pp. 459–466, 2005.

while taking care of constraints. The uncertainties about the environment and the effects of actions, the environmental constraints, goal interactions, and time and resource constraints make the problem harder. Therefore, flexible search strategies should be designed to make the problem more manageable. Time is a very hard constraint while finding the humans suffering from the disaster. There is a tradeoff between effective planning without dead ends and the fast plans for finding the victims in the environment. Partial-planning or re-planning [5] strategies are known to be very effective. However the use of either of them as such is not suitable for real implementations of the SR robots. However, key ideas of both approaches could be used in the design of a planning layer design of a SR robot to take advantages of both. Architectures such as three layered or BDI (Belief-Desire-Intentions) may be used to implement in the planning layer design depending on the specific application [8].

Since the difficulties with the environment make the current SR robot prototypes to be designed as semi-autonomous or highly dependent on human robot interaction, autonomous robot designs are rarely encountered in the literature [1]. In this study, an autonomous planning strategy suitable for the environments when the aid of human rescuers or search dogs is completely impossible is proposed. The work area for a robot running this planning strategy can be small voids or possibly dangerous environments with gas leakages or under danger of possible explosions or further collapses. The layered model of InteRRap and the belief update procedures of BDI type architecture are combined, and exception handling strategies are attached to the proposed planning layer action selection mechanism to cope with the uncertainties on the environment. Reactive actions are executed directly according to the sensory information. This proposal of the hybrid planning strategy is believed to be useful in promoting further improvements in SR robot designs.

The rest of this paper is organized as follows: Section 2 introduces the proposed planning module design and the proposed strategies. In Section 3 the experimental results are given. Finally, Section 4 presents future work and concludes the paper.

2 Search and Rescue Planning Module Design

2.1 Requirements

The primary goal of the Planning and Behavior Module (PBM) of a SR robot is to find victims in the disaster area within a minimum possible time interval. The secondary goals are avoiding obstacles, avoiding risking resources or triggering a further collapse, searching the area effectively, avoiding cycles in the search, turning and moving to the directions of locations in which it is believed that humans to be rescued are located, turning and moving to the directions of locations in which it is reported by other robots or a dispatcher that humans to be rescued are located.

The PBM of a SR robot should be able to determine a plan based on the current state of the environment, the robot's location, the hierarchical structure of the desires (goals), and the constraints. It should also perform re-planning

when necessary while executing a plan based on the intermediate changes on the environment or the internal state.

There may be static or dynamic obstacles, other robots, and the human workers of the rescue team in the environment. The environment structure and the map are unknown.

2.2 Design

The proposed PBM design consists of mainly three layers which interact with each other to achieve the same end as in the InteRRap architecture. The reactive layer interacts directly with the sensory modules, and produces an appropriate action. The planning layer constructs the plan for the robot to implement its task in an optimal way. The strategies for avoiding cycles, forming beliefs to direct the search space, and exception handling are implemented by this layer. The communication layer is the interaction layer of the module and implements the robot communications. It is responsible for informing other robots according to the planning layer outputs, and receiving the incoming information. The bandwidth requirements are very small.

PBM interacts with other modules of the robot to make decisions and to convert these decisions into actions. The actions based on the plans produced by the PBM are sent to the motor interface unit of the robot.

In the design, the following assumptions about the environment are made:

- The assigned rubble sub-area for the robot is a grid like environment.
- Robot's visibility space contains 8 current neighbor cells.
- The corner neighbor cells cannot be reached if the neighboring cells are obstacles
- The obstacles are modeled as gaps or untraversable huge obstacles
- The area surrounded by untraversable obstacles is not considered

The only assumption for the architecture of the robot is that it is battery powered and equipped with some special simple sensors, and it can operate several hours.

In the design of the PBM, effective algorithms for both exploration and exploitation are proposed. SR robot works in a dynamic and unknown environment. Initially, the only information available to the robot is the preloaded knowledge-base. In the main loop of the planning strategy, if there is some information about the possibilities of victims, and their possible locations, this information is used. Otherwise the area is explored effectively. In this manner, the strategy allows searching of the locations farthest from the visited locations to explore unknown locations of the victims. This loop continues whenever the state is recoverable and it is believed that there may be victims in the area. During search, the obstacle avoidance is implemented before collisions occur.

2.3 Planning Strategy for Victim Locations

In the strategy for moving to the locations believed to contain victims, the shortest path is considered. Therefore the robot initially turns to the direction

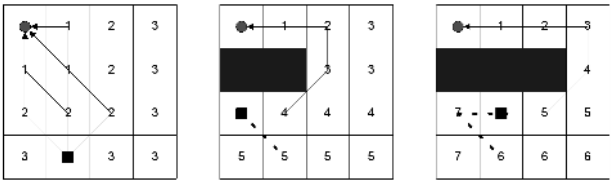

Fig. 1. The update strategy of cell distances to the target on the planned path

of the victim and tries to move forward continuously. Exceptions can occur when the robot encounters an obstacle. It can determine whether the object is dynamic or static by means of sensory inputs. Since the proposed approach is suitable for all SR robot architectures including ones lacking of such sensory equipments, the object type test is implemented blindly by waiting for a short time. If the object still stands in front, it is assumed to be a static object. If the prediction is wrong, in later steps, this information can be corrected because of the adaptive world knowledge update mechanism. For more advanced architectures having efficient sensors, this test process is not needed. During search, alternative directions should be tried. After alternative path selection, the shortest victim location choice is again executed. The *Locating Human Beings* Module may indicate another victim location closer than the target. Such information can also be sent from another robot. In this case, the target is changed, and the closest victim is chosen to rescue to satisfy the primary goal. When the target is changed, the information related to the previous target is sent to the dispatcher so that other robots can save the victim.

If there are no more targets, and it is believed that nobody is alive in the assigned sub-area a *finished* message, or if the situation cannot be recoverable, a *help* message is sent to the dispatcher. If the deadline ends before reaching the target, this information is sent to the dispatcher, and a new target is chosen.

The nearest victim path-planning strategy provides the robot to traverse a path, visiting victim places by using the shortest path. The predicted cell distances to the selected victim target are updated at every movement, and also when the world knowledge is updated as in Free Space Assumption planning theory [4]. An illustrative example of the update strategy of cell distances on the planned path can be seen in Fig. 1, where, the square represents the current location, and the circle the target. The lines with arrows are the alternative paths. The dotted line is the path taken by the robot. The numbers in the cells are the guessed distance values to the target. Initially the environment is completely unknown, assuming the sensor range is one-cell distance. After updating the world knowledge, the observed obstacle information is taken into account to predict the distance values close to real, and to provide an incremental search strategy.

The probability information for victim locations is stored as a vector in the long-term-memory. If the robot finds a victim in a location, it clears the probability information, and adds the location to the rescued location vector and sends a *saved* message to the dispatcher. Such a message can also arrive at the

```
NearestVictimPathPlan
  do while there is a victim probability
    ChooseClosestVictim;
    UpdatePathInformation;
    if a path to the target is not available
      Assign this target as unreachable;
      Send a declaration message to the
      dispatcher informing the victim location;
    else
      DefineDeadline;
      SelectClosestRotation;
      Move;
      if PlanFails
        HandleErrors;
    update worldKnowledge;

UpdatePathInformation
  from target expanding to all neighbor cells
    assign ∞ for the current distance;
    choose the smallest neighbor distance
    and update the own;
HandleErrors
  if there are obstacles on the path
    UpdatePathInformation;
    turn to the cell closest to the target
  if more than one cell with the same distance value
    choose the least visited;
  if the situation cannot be recoverable
    send a help and go to the idle mode;
```

Fig. 2. Moving to the nearest victim strategy

communication layer of the current robot. If the location in the message is the current target, the robot clears the probability information for this location and selects another target location in the next step. The obstacles information is also stored in the long-term memory. However since the environment is dynamic, the planning strategy takes into account of local changes while moving. In this case, the information related to the obstacles is updated. Therefore the wrong beliefs related to the dynamic obstacles labeled as static are corrected. The searching strategy algorithm is given in Fig. 2.

2.4 Search Strategy for Exploration

When the robot has nothing to do better, it tries to search the environment in an effective manner. This situation occurs when there is no motivation about a possible location believed to contain victims.

A selective pressure value is defined for unvisited locations, which is updated for each movement. The added value to the previous pressure value is a function of the distance between the cell and robot's location, and the number of unvisited neighbors of the cell. A pressure value of a cell is updated based on Eq. 1

$$pr_i^{k+1} = \begin{cases} 0 & \text{if } i \text{ is within the sensor range} \\ pr_i^k + dist(i(x,y,z), R(x,y,z)) + 10 * n_u/n_t & \text{otherwise} \end{cases} \quad (1)$$

where pr_i^k is the pressure value of the cell i at step k, n_u is the number of unvisited neighbors and n_t is the total number of neighbors in the sensor region, $dist()$ returns the Euclidian distance between two points in 3D, $i(x, y, z)$ is the cell location, and $R(x, y, z)$ is the robot location. The selective pressure values of cells within the sensor range are set to 0.

In the exploration strategy, a cell having the highest selective pressure value is selected as the new target. After selecting the target, a path is planned containing maximum number of unvisited cells. For the current cell, the three neighbors closest to the line between the current cell and the target cell forms a sub-region. An unvisited cell from this sub-region is selected. If there is not an unvisited sub-region neighbor, the other neighbors are examined and the unvisited neighbor

```
LeastVisitedLocationPlan
  Choose unvisited location with the
  highest pressure value;
  do while (target is not reached)
    if there is victim information
      break;
    UpdatePathInformation;
    ChooseTheSubRegionNeighbor;
    Move;
    UpdatePressureValues;
    if PlanFails
      HandleExplorationErrors;
    update worldKnowledge;
HandleExplorationErrors
  if there are obstacles on the path
    UpdatePathInformation;
  if short term memory indicates that
  the situation cannot be recoverable
    send a help and go to the idle mode;
  if a path to the target is not available
    Assign the target as unreachable;
```

Fig. 3. Exploration Algorithm

closest to the target is chosen. If all the neighbors are visited before, the sub-region cell having the minimum visit value is chosen finally. The visited locations are removed from unvisited vector. As soon as any information about a victim location is determined, the nearest victim path-planning strategy is switched immediately. The proposed exploration algorithm is given in Fig. 3.

3 Experimental Results

A simulation environment in C++ using OpenGL Library was implemented to measure the performance of the proposed algorithms. The environment is constructed with random locations of obstacles and victims. The dynamic objects representing the refugees or human team members are allowed to walk or run in the environment. The number for sensor range indicating the maximum cell distance to sense any victim probability can be adjusted.

Since there is no similar algorithm, the designed algorithm was only compared with the greedy mapping method. Mine sweeping, vacuum cleaning or lawn moving planning approaches [1] could also be selected for comparison. In these tasks the cells should be visited at least once. However these algorithms are designed without time constraints whereas in SR case this is crucial.

The Greedy Mapping Method [4] is an adaptive method running on an unknown environment. While selecting the shortest unvisited cell in each step, the method traverses the shortest path from initial state through the goal state. The proposed nearest victim path-planning strategy is combined with this strategy to conduct tests.

The experiments were conducted with randomly generated 20x20 grids or mazes. The proposed strategy provides a complete search with an effective exploration capability. The exploration time (Coverage) or the number of steps to find victims (Goal) change based on the number of obstacles and the structure of the environment. The results of the proposed strategy can be seen in Table 1. Each row represents different obstacle densities.

As expected, the number of victims found decreases when the number of obstacles in the environment increases. The exploration time to visit all of the

Table 1. The performance of the proposed strategy for different densities

Real Obstacles		**Detected Obstacles**		**Coverage**		**Goal**		**Number of Victims rescued**	
μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
56.04	7.10	56.16	7.15	531.72	40.31	119.6	50.96	4.92	0.28
122.40	8.81	129.2	13.06	548	103.78	171	61.97	4.9	0.31
199.80	10.12	237.90	24.62	341.70	96,05	102.90	56.60	2.80	1.48

Table 2. Comparison of the exploration strategy with greedy mapping method

Sensor Range	**Greedy Mapping Exploration**				**Proposed Exploration**			
	Coverage		Goal		Coverage		Goal	
	μ	σ	μ	σ	μ	σ	μ	σ
2	411.52	23.97	270.28	72.22	563.76	41.42	173.64	80.44
3	407.44	16.07	266.68	50.97	540.88	37.46	92.62	43.38
4	402.08	13.50	231.96	71.46	518.72	26.86	65.68	31.10

cells at least once decreases while obstacles density increases. The numbers of real and detected obstacles are different because of unreachable places formed by the combinations of the obstacles. Although it can be inferred that as the number of obstacles increases the number of steps to find the victims decreases, the number of victims rescued decreases drastically due to the unreachable places containing victims.

Table 2 presents the results for the exploration strategy for both the proposed and the greedy mapping exploration methods. The proposed nearest victim path-planning strategy is applied for both methods. The same randomly generated grids are used to compare the methods. The number of victims in the environment is 5 and the average number of cells containing obstacles is 29.54 with standard deviation 5.39. The results are average of 25 independent runs for each sensor range. The algorithms are compared for both the number of coverage steps (Coverage) and the number of steps to find all reachable victims (Goal).

It can be noted that the number of steps to find the victims in the environment decreases when the robot's sense region is increased. Therefore this result shows that the selection of the specialized sensors is very important for SR mission. As can be seen from the table, for the proposed exploration method the number of steps to reach the goal is smaller than that of greedy mapping. However the number of steps to cover the whole reachable places is greater. But for SR operation, to reach the goal in a minimum possible time is more important than covering the area with a smaller number of steps.

4 Conclusions

In this work, a novel autonomous planning strategy suitable for all types of SR robot architectures is proposed. The planning layer design can be used in

the environments where the aid of human rescuers or search dogs is completely impossible. The conducted experiments show that the proposed strategies are complete and promising for the main goal of a SR robot. The number of steps to find the reachable victims is considerably smaller than that of the greedy mapping method. In the proposed exploration strategy, the environment is explored to find the victims as soon as possible by selecting the least visited cell as a new target. Although, the proposed strategy completes a single coverage of the environment with greater number of steps, this is acceptable due to the nature of the SR mission. The environment is searched till it is believed that nobody is still alive in the area. While visiting the least visited neighbors on the way to the target, the environmental changes can also be tracked.

As a future work, 3D real life experiments with real robots in a disaster area should be implemented and extending the ideas given in this paper to the multi-agent case can be considered.

References

1. Casper J. and Murphy R.R.: Human-Robot Interactions During the Robot-Assisted Urban Search and Rescue Response at the World Trade Center, *IEEE Transactions on Systems, Man an Cybernetics-Part B*, Vol. 33, No.3 (2003)
2. Davids, A.: Urban Search and Rescue robots: From Tragedy to Technology, *IEEE Intelligent Systems*, Vol. 17, No. 2, (2002) 81-83
3. Kitano, et al.: RoboCup-Rescue: Search and Rescue in Large Scale Disasters as a Domain for Autonomous Agents Research, *IEEE Conf on Man, Systems, and Cybernetics* (1999)
4. Koenig S. et al.: Greedy Mapping of Terrain, *Proc. of the International Conference on Robotics and Automation,* (2001) 3594-3599
5. Murphy R. R.: *Introduction to AI Robotics,* England: The MIT Press, (2000)
6. Murphy R. R.: Marsupial and Shape-Shifting Robots for Urban Search and Rescue, *IEEE Intelligent Systems*, Vol. 15, No. 2, (2000) 14-19
7. Takahashi T. and Tadokoro S.: Working with Robots in Disasters, *IEEE Robotics & Automation Magazine,* Vol. 9, No. 3 (2002) 34-39
8. Weiss G.: *Multi Agent Systems: A Modern Approach to Distributed Artificial Intelligence*, England: The MIT Press, (1999)