Rescue Simulation League - Virtual Robots Pasargad Team Description Paper

Abbas Razzaghpanah, Arash Nouri, Fatemeh Hemmati, and Yaser Abbasi

Mathematics and Computer Science Department, Amirkabir University of Technology, Tehran, Iran
{r.panah,arash1718,fateme.hemati,yas.abbasi}@aut.ac.ir

Abstract. The team started in the summer of 2010, in hope for providing enhanced coding aid for the Pasargad Real Rescue Robot team. Since, the main goal has been set to make a useful environment for implementing, as well as testing algorithms such as SLAM and obstacle avoidance algorithms. This paper is going to briefly describe the work done so far to achieve that goal.

1 Introduction

Pasargad Real Rescue Robot team have a notable reputation in robotic competitions. In Pasargad robotics club, a group of students in fields of computers, electronics and mechanics have gathered to develop robotics in various aspects. The club consists of two virtual and real rescue robotic teams that collaborate in both types of competitions, forming a joint team.

The computer programming team in both teams are a unified group trying to import real robots in the USARSim simulation world to decrease time spent testing the real robots and thus, reducing development costs. More in [1].

So far, the virtual robot team have created a map close to what's used for real robots and is currently modeling the real robot in the simulation environment in order to simplify implementation of localization, mapping and obstacle avoidance for the real robot by simulating them in a virtual world.

The team have had advances in teleoperation and is looking forward to use joypads [2] and voice control commands as a means of robot control in future events.

2 Localization and Mapping

In today's world, robots have to operate in environments and terrains that may or may not be known. In order for a robot to navigate through such places, a simple map of geometry and obstacles should be generated and the robot's location within that map should be estimated. And this, has to be done using sensors and cameras mounted on the robot. The SLAM algorithm can produce a low-error map with the estimation of robot's placement and direction based

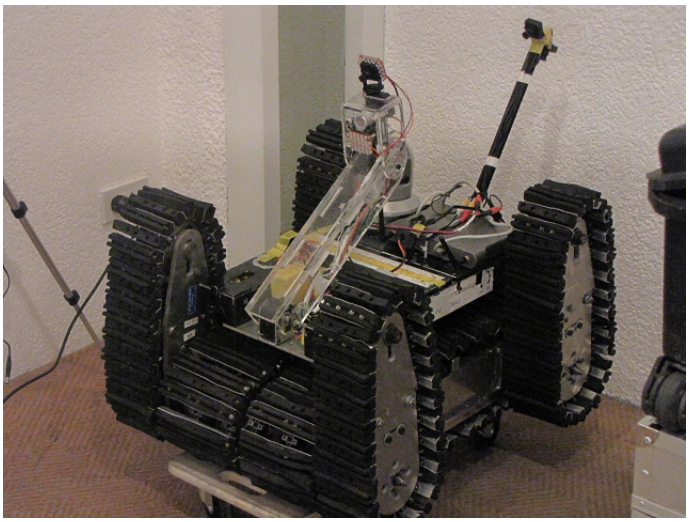

Fig. 1. Pasargad Teleoperated Rescue Robot

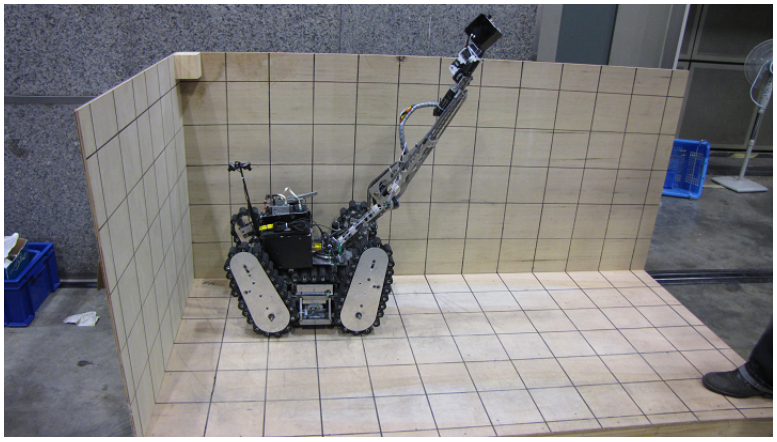

Fig. 2. Pasargad Teleoperated Rescue Robot in Action

on data streams from sensors that are updated as the robot moves. Topology-based maps can be automatically extracted from an occupancy grid built from sensor data using techniques borrowed from the image processing field. Since these techniques can be soundly defined on fuzzy values, our approach is well suited to deal with the uncertainty inherent in the sensor data. Topology-based maps are fairly robust with respect to sensor noise and to small environmental changes, and have noise computational properties. [3]

Topological mapping and metric mapping are the main approaches to obtain a map. The topological map consists of nodes that represent the significant spaces in the environment, and edges that represent the connection between the spaces. For this reason, the topological map is compact and suitable for global path planning. the topological map makes the cleaning robot simply perform room-to-room path planning by dividing the environment. [4]

Virtual door is defined as all possible locations of real doors in an occupancy grid-map, and the occupancy grid-map is obtained in orthogonal environment. Virtual doors are detected by extracting corner features from occupancy grid-map, and an initial topological map is generated which has many nodes (rooms) and edges (virtual doors). The final topological map is generated by using a genetic algorithm (GA) to merge the nodes. [5]

GA is suitable for complex problems or poorly understood problems that have

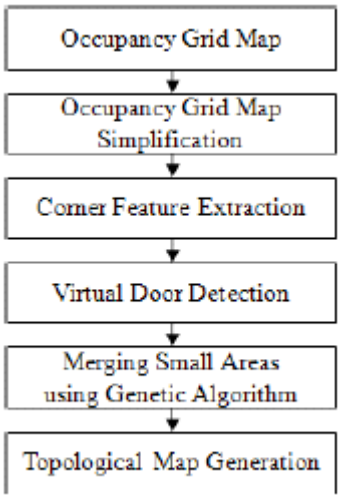

Fig. 3. Mapping Steps

large search space. For example, if twenty virtual doors are detected, the problem has 220 solutions in the search space because each virtual door has two options: removed or not. In the case of exhaust search, all solutions need to be evaluated,

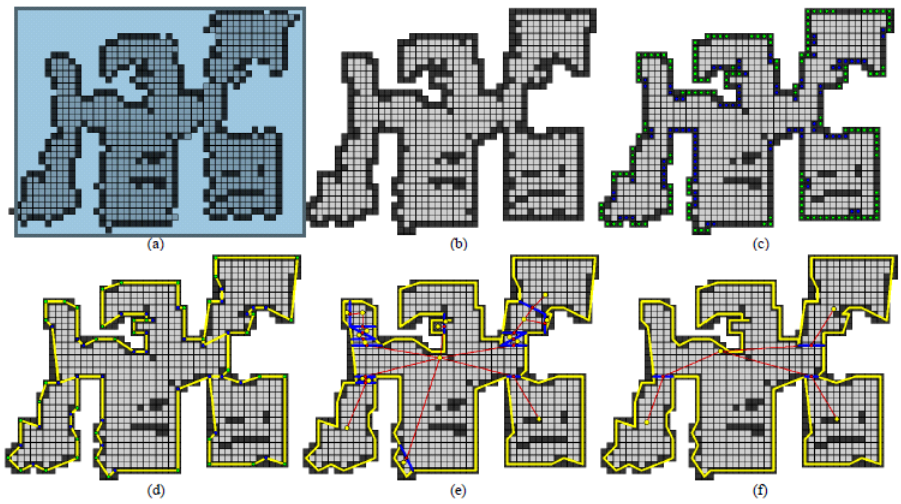

Fig. 4. Green circles: concave features; blue circles: convex features; yellow circles: nodes located in centers of rooms; red circles: edges located in centers of virtual doors; yellow line: boundary of indoor environment; blue lines: virtual doors; red lines: connections between each edge and neighboring nodes. (a) Occupancy grid-map, (b) simplified occupancy grid-map, (c) sampling of obstacle grids, (d) corner feature extraction, (e) initial topological map, (f) merged topological map.

this task has a large computational load, and is not time-efficient. GA can be used to decrease the computational load and time for the optimization of the problem. A detailed description of proposed methods are described in [5].
A robot bases it's decisions on the place in which it is located on the map. In order for the robot to localize, it uses absolute and relative measurement data of it's driving actions and a rough image of it's surroundings. The "Kalman filters" is the algorithm used to localize the robot by considering all possible mappings and robot placements and picking the ones that are most likely the real ones. [6]
A fixed laser sensor mounted on the robot can produce error in mapping since the robot may misestimate distance while treading on inclined surfaces. To overcome this, a stabilizer is used to keep the laser sensor horizontally level.

3 Exploration

In virtual competition all of teams try to explore with SLAM. It means that every robot must search environment, localize and generate map at the same time. With this action each team can optimize its performance. Exploration has two important steps:
1- Victim Detection
2- Obstacle Avoidance
Both of these steps could be done with sensors and actuators.

3.1 Victim Detection

The most important part of a rescue robot's job is finding and recognizing the victim in robot's observations in its exploration. We deployed different types of sensors for victim detection (i.e. sound, touch, human motion and victim-and-false-positive sensors).
Our algorithm is implemented considering this probability that using only the victim-and-false-position sensor or touch sensor may lead to false detection or causing latency in finding victims. Every victim may have motion or make sound, so it is possible that human motions or sounds help us to find victims. One of the benefits of using sound sensor in virtual environment is that, there is only one source of sounds which is made by victims, so it can help each robot to detect victims with better accuracy.
Another perception of a victim is the pictures that the robot's camera captures.These pictures denote the motions and natural colors of the environment. Our program uses the data which comes from camera to recognize the color of the skin or the shape of the victim's face or body. Skin color is an important parameter in robot's vision to detect victims. Given skin and non-skin histograms, can increase the proba-bility of finding victims in the area which is a part of robot training. The method we use to detect the color of skin is based on the YCbCr color space.

3.2 Obstacle Avoidance

Obstacle Avoidance is one of the major problems in autonomous robot exploration, in which in, robot uses sensors' data and makes a decision to choose the best trajectory. Obstacle avoidance is used when the agent is located where in a manually inoperable area. For example, when the robot is out of communication range, agent must explore an unknown environment, and yet, shouldn't strike into any available obstacles. Here, it would be good to mention that we cannot call all the substances which robot senses, the obstacle. In our agent architecture definition; walls, blocks and big hedges are not obstacles because they are part of the global map and could be seen prior to reaching. On the other hand, an obstacle is a hedge which can be sensed with robot's sensors (in our agent architecture we use sonar sensors for obstacle avoidance) and makes the robot change its trajectory to reach its goal or target. Obstacle avoidance focuses on changing the robot's trajectory as informed by its sensors during motion. The resulting motion is both a function of the robot's current or recent sensor readings and its goal position and relative location to the goal position. In our agent architecture, obstacle avoidance is a low-level module which uses sonar data to guide the robot away from obstacles. When the robot comes too close to an obstacle, it overrides the control commands sent by the navigation module, and drives the robot away from the obstacle. We first used Bug Algorithm for avoiding any obstacles. Then we worked on Vector Field Histogram (VFH) [7]; therefore, we reached to this conclusion that it is better that we apply a compound of that two methods and algorithms. Therefore, in our compound algorithm we define a Cost function that uses from sonar data for selecting the best trajectory. In this algorithm our robot select the best way for avoiding any obstacles with maximum cost values for going towards its goal.

4 Connection Management and Multi-Client Control Handling

Regulating and managing control client (or clients) necessitates an expert connection manager that can supervise several streams of commands and state-data stream transmission (it could also conduct several teleoperators to communicate with the same connection). This data manager could also directly handle image data in order to preserve bandwidth for decision server.
A proxy server should be implemented to manage data flow between clients and the decision server, and another should be in charge of communication of the decision server and the robot interfaces. The latter could also come in handy as layer interfacing with the server when rules and command regulations are subject to frequent change. In addition, this method makes offline debugging and testing easier since the proxy server can simulate a given condition off a log file, as many times as desired. Offline debugging is the act of tracing the exact set of events to find the problem that's caused by those events. In order to do that, proxy server could provide a pre-recorded feed for the decision server to

simulate the circumstances under which the problem procures, in order to ease debugging.
Having stated its importance, we've considered implementing two proxy servers for use with our agents. The diagram below schematically traces what we have in mind.

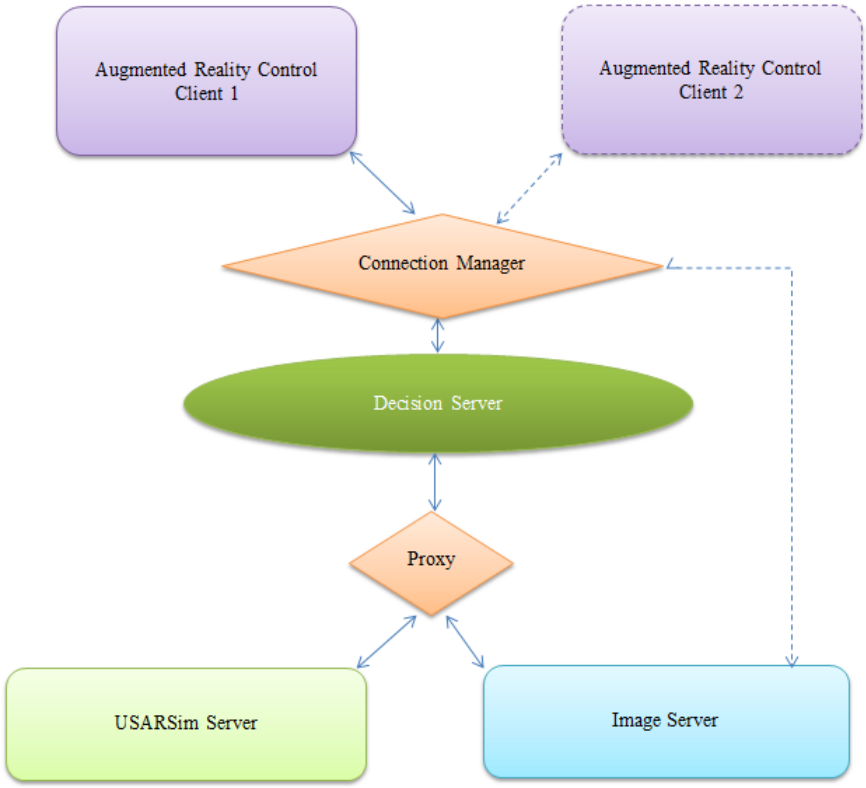

Fig. 5. Communication Structure

5 Future Aspects and Real World Applicability

Pasargad virtual robot team would like to participate in the 2011 International Robocup Event in Istanbul and bring the collaboration between the real robot and virtual robot teams to a whole new level.

References

[1] Jijun Wang, Stephen Balakirsky, Stefano Carpin: USARSim: a robot simulator for research and education

[2] Clayton Walnum: Teach yourself Game Programming with DirectX, SAMS
[3] E. Fabrizi and A Saffiotti: Extracting Topology-Based Maps from Gridmaps, International Conference on Robotics and Automation, vol. 3, 2000, pp. 2972-2978.
[4] E. Fabrizi and A. Saffiotti: Augmenting topology-based maps with geometric information, Robotics and Autonomous Systems, vol. 40, (2002), pp. 91-97.
[5] Kwangro Joo, Tae-Kyeong Lee, Sanghoon Baek, and Se-Young Oh: Generating Topological Map from Occupancy Grid-map using Virtual Door Detection, IEEE.
[6] Professor Rick Middleton: Applications of the Kalman Filter Algorithm to Robot Localisation and World Modelling, Electrical Engineering Final Year Project, University of Newcastle, NSW, Australia, 2002.
[7] J. Borenstein, and Y. Koren: The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots, The University of Michigan, Ann Arbor, Advanced Technology Laboratories, 1101 Beal Avenue, Ann Arbor, MI 48109