Robocup Austria 2009 –Rescue Simulation League Virtual Robot Competition ETB_VR

Majid Yasari, S. Ali Zahiri, Sepehr Farhand

ETB Robotic Association Labs, Computer Engineering Department
East Tehran Branch Islamic Azad University, Tehran, Iran
{majid.yasari, sa.zahiri, farhand.sepehr}@gmail.com

http://www.etbiau.ac.ir/robotics.htm

Abstract. This article is a quick overview of design decisions made by the ETB Virtual Robot team for participating in Robocup 2009 Competition in Austria. We are going to give a short description of our agent architecture and the design of our Robot and techniques that are used in order to deal with several challenging problem in Virtual Rescue Robot such as SLAM (Simultaneous Localization and Mapping), Autonomous Exploration, Obstacle Avoidance and Victim Detection.

1 Introduction

Every year in all around the world, thousands of people die or get injured under collapsed buildings or other unreachable areas because of explosion, earthquake, flood and other natural or unnatural disasters. Delay in rescue operations due to these dangerous situations, vast locations, massive casualties, etc. leads us to employ high technologies to aid rescue group.

Designing, implementing and using rescue robots not only makes explorations much more accurate by seeking for vital signs of victims, but also brings much more safety to rescue teams.

Because of mentioned problems, shortage of equipments, cost overheads and also to prevent doing several experiments repeatedly which takes a lot of time, nowadays simulated environment are commonly used. Urban Search and Rescue Simulation (USARSim) is an environment which is used in robotics and artificial intelligence methods.

Since 2002, RoboCup Rescue Competitions are held as a part of the annual RoboCup World Championships. It is the purpose of RoboCup Rescue, to promote research and development in this socially significant domain in order to ultimately acquire solutions that can be used by USAR (Urban Search and Rescue) teams under real emergency situations.

In this paper, we present a system of heterogeneous team of semi-autonomous robots, which explores an unknown environment. The tasks are performed coopera-

tively to ensure maximum information extraction of the environment, in order to assist first responders to plan rescue operations.

Our agent architecture is designed to use intelligent function up to fully autonomous [1]. This includes works on autonomous multi-robot systems. For examples, mapping a vast area with multiple robots [2] and research on exploration under constraint of wireless networking.

We use some principle algorithms such as SLAM Algorithm to localize and map the environment, this algorithm is one of the main algorithms for localizing the agent and mapping the environment simultaneously; in resumption, we use compound of Bug algorithm and Vector Field Histogram (VFH) for preventing to strike any obstacles and barriers which maybe are between our agent and target.

2 Localization, Autonomous Navigation and Mapping

Intelligent robot needs to use SLAM techniques to explore and rescue in real or virtual environments. There are two approaches in SLAM:

1) Where am I?
2) What is the scale and shape of my environment?

Answer to the first question solves localization and answer to the other one solves Mapping**.** So SLAM is a technique that is employed by robots to design a map and create a world model from the unknown environment. SLAM implementation includes several steps. The data which robot receives from its sensors are fully noisy and it defects the performance considerably. So first of all we should make the data smooth and eliminate noise from it. We proposed Kalman filter as a good solution for it [3, 4 and 5]. On the other hand, decomposition of the map into several sub-maps is another approach; this method called occupancy grid, finds landmarks in the main map and then creates grids on it.

$$\overline{X}_t = X_{t-1} + MA_t \qquad \textbf{(1)}$$

$$\overline{Y}_t = Y_{t-1} + Y_{actuator} \qquad \textbf{(2)}$$

In (1) and (2) the map has updated due to the linear motion;

- A_t : store the current motion commands.
- $Y_{actuator}$: denote the associated uncertainly.
- Z_t : Current looking
- K_t : gather the estimate distance.

$$K_t = \overline{Y_t}\ C^t\ (C\ \overline{Y}_t\ C^t + Y_{measure}) \qquad \textbf{(3)}$$

$$X_t = \overline{X_t} + K^t (Z_t - C \overline{X_t}) \quad \textbf{(4)}$$

$$Y_t = (I - K_t C_t) \overline{X_t} \quad \textbf{(5)}$$

In (3, 4, 5) X_t , Y_t is the result that get from Kalman filter and X_{t-1}, Y_{t-1} is the input data for Kalman filter.

Occupancy Grids have some benefits; the most important one is that grid-based representations of a robot's physical environment can be used directly by most navigation, obstacle-avoidance and learning algorithms. Another key benefit of occupancy grids is that the resolution can be tuned to represent the environment's geometric properties at any desired amount of details. In theory, this is bounded above only by the level of details originally captured by the sensors. This property of occupancy grids makes them the ideals metric representation for maps that should contain a high amount of detail, a feature that is not used in most other map representations. [6, 7].To accomplish occupancy grids, there are sensors in the robot that use a simple technique with ray-casting .This beam has been sent directly from the robot to obstacle to determine distance between robot and obstacle.

For example, if an obstacle is detected at some relative distance, then rays can be cast from the cell at the current position of the robot towards every cell that coincides with the detected obstacle. For all the cells which intersect with the cast rays before the rays hit on the obstacle, the miss counter is incremented. Likewise, for the cells that intersect with the detected obstacle the hits counter is incremented [8]. Afterwards, the occupancy of every grid cell can be determined by threshold of the ratio of the number of hits over the sum of hits and misses together (6).

$$\text{Occupancy of each cell} = \frac{Hits}{Hits+Miss} \quad (\textbf{6})$$

After occupancy grids, another approach for localization will be used, which is based on Topologically Organized Map. In this step a special graph will be created from the path of robot. In addition knowing that the obstacle is detected, a diagram will be created from the map to divide it into the sub-maps. These sub-maps have no overlap with each other and each one describes a safe area for robot to move in that region [9].

The first advantage of using topologically map is its outstanding support for path planning algorithms. Autonomous robots need to go from a place to another in order to complete their designated tasks. Considering the task of exploration which all robots need to perform, it is conceivable that in some points the robot reaches a dead-end. In this situation before it can continue to search another area, the robot has to get out of that location first. Topological maps facilitate these kinds of tasks very well. The nodes and links in graphs and diagrams tell any path planning algorithm precisely how to go through the explored areas safely.

The second advantage is their compactness. They are capable of representing huge environments in a very compact way. While the size of occupancy grids and feature-based maps grow exponentially with the size of the environment or the number of

detected features, topological maps typically only grow linearly in size as nodes and links are added to denote recently explored areas.

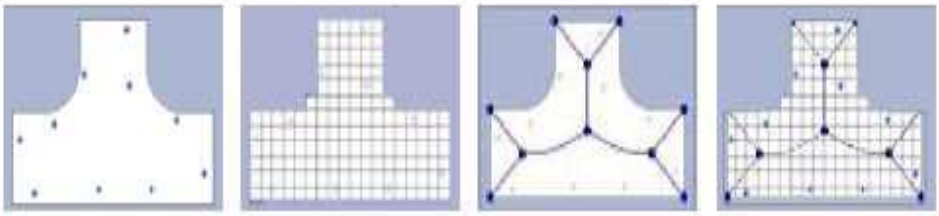

Fig. 1. Applying four steps for localization and mapping [8]

So as Fig.1 shows, we can summarize the steps of mapping and localization;

- Applying Kalman filter on feature based map.
- Making a grid cells from the map.
- Implement topological map to determine obstacles and safe paths.
- Decompose the main map and start mapping from unknown environment and design a new map.

So, each robot performs exploration, mapping and localization with the algorithm described above and creates its own sub-map. Finally, all of the maps which are separately designed merge with each other to achieve the main map (7).

$$\text{Main Map} = \text{Merge}(\text{Sub-map}_1, \text{Sub-map}_2, \ldots, \text{Sub-map}_n) \quad (7)$$

Using topological map we have required information for planning, there are some methodologies proposed for planning regarding; A* algorithm [10] and other AI approaches, heuristics, classic algorithms based on graphs i.e. Shortest Path Algorithms (e.g. Dijkstra, Floyd), Graph Traverse Algorithms (e.g. BFS, DFS, etc.) In our limited time we found our heuristic algorithm much more efficient both in functionality and amount of processes.

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 employed 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 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 more 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 probability 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 which in it, robot uses sensors' data and then makes a good decision to choose the best trajectory. Obstacle avoidance is used when our agent is located where we cannot operate it manually with an operator. For example, when our robot is out of range of communication station, agent must explore an unknown environment, and yet, shouldn't strike to any available obstacles. Here, it is 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 global map and could be seen before reaching them. On the other hand, obstacle is a hedge which can be sensed with robot sensors (in our agent architecture we use Sonar sensors for obstacle avoidance) and makes robot changes its trajectory for reaching to its goal or target. Obstacle avoidance focuses on changing the robot's trajectory as informed by its sensors during robot motion. The resulting robot 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 the sonar data to guide the robot away from obstacles. When the robot comes too near an obstacle, it overrides the control commands sent by the navigation module, and drives the robot away from obstacles.

We first used Bug Algorithm for avoiding any obstacles. Then we worked on Vector Field Histogram (VFH); 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 Communication

In virtual robots like other simulation environments there are two main parts; the agent and the server. The agent represents one or more robots with sensors and actuators which get sensed from and affect the virtual environment. Fig.2 describes the way server and robots communicate with each other.

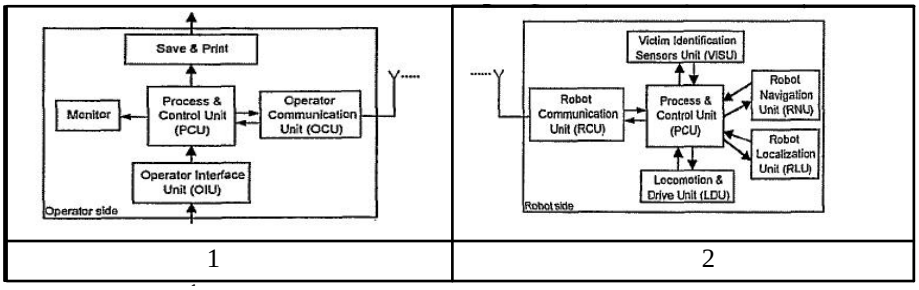

Fig. 2. The architecture of server side (1) and client side (2) of connection

This decentralized setup allows us to spread the computations across many machines and still run a relatively large team of agents. Only one server instance is required, which will record the pre-processed mapping information from the agents.

Robots use wireless communication in order to share some information about the region they are exploring. Although this communication is limited by distance, but sometimes it helps robots to minimize overlapping in explored areas and improves coordination in multi-agent fields.

5 Agent Architecture

Agent Architecture is the frame of all algorithms used in this article. So without strong agent architecture, we cannot develop a stable system in which it could perform its tasks as good as possible.

The ETB rescue virtual robot team intends to field a team of robots to perform coordinated mapping, exploration, and victim detection. However, each robot must also be able to perform its tasks individually and does not rely on other robots. For this purpose, we have designed a modular system with each module being as self sufficient as possible. Also, each module can be changed internally without affecting the other modules.

This solution helps us to insure the stability of the system and prevent any possible crashes. All incoming and outgoing messages are parsed via one frame so all connections between the server and the agent will be handled by one class so any changes in server configuration can be handled easily by only changing this frame. Our agent architecture can be divided into the following modules and below we demonstrate relations between modules of our agent architecture:

1. Operator Interface
2. Communication
3. Exploration
4. Victim Detection and Obstacle Avoidance
5. Localization, Autonomous Navigation and Mapping
6. Sensor and Actuators (and location of them on robot's body)

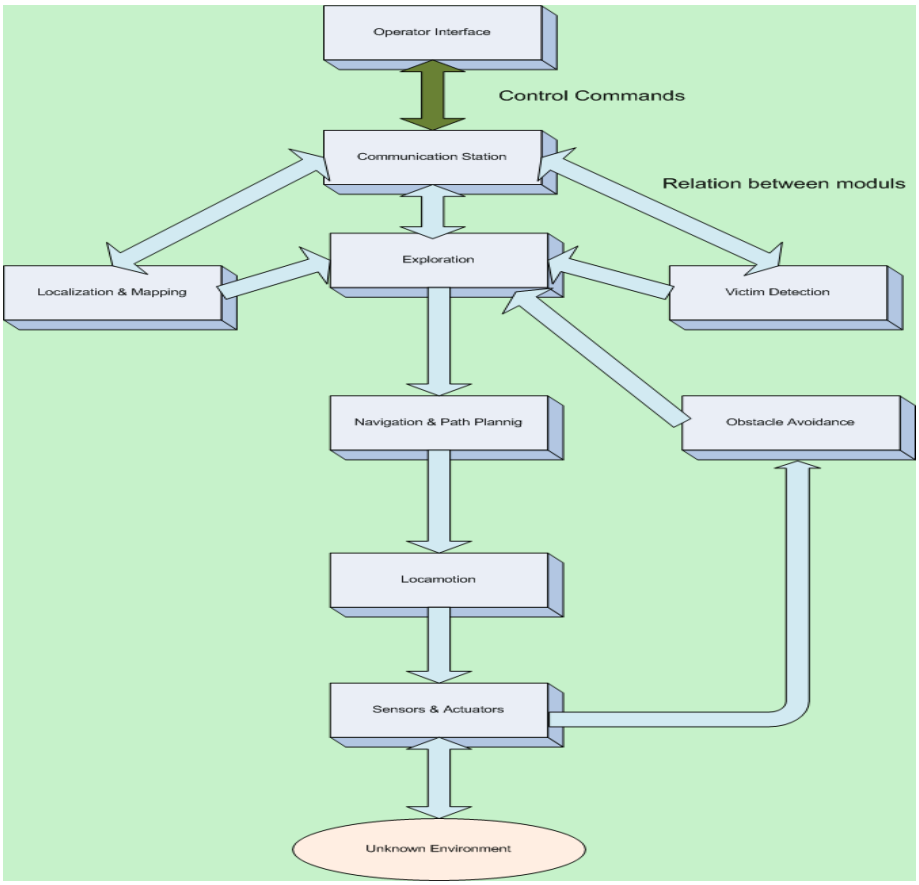

Fig.3 Relations between modules of our agent architecture

6 Conclusion and Future Decision

This paper has discussed goals and features of ETB simulated virtual agents, including detailed specifications and algorithms. Our team consists of 5 people who 2 of them have the experience of participating in RobpCup competitions; however this is the first time that ETB is participating in Virtual Robot Competitions. For this competition we tried to test our team and obtain positive and negative side of our team code and develop it. Also, we demonstrated SLAM, exploration, communication and some basics were introduced. Furthermore, some applicable approaches and useful algorithms are presented to discover the unknown world around. We hope to implement fully autonomous agents with distributed decision making system which can give a proper map in short time and more accuracy by developing our SLAM algorithm and more reliable victim detection skills.

References

1. Birk, A., Kenn, H.: control architecture for a rescue robot ensuring safe semi- autonomous operation. In Kaminka, G., U. Lima, P., Rojas, R., eds.: RoboCup-02: Robot Soccer World Cup

2. Birk, A., Carpin, S.: Merging occupancy grid maps from multiple robots. IEEE Proceedings, special issue on Multi-Robot Systems

3. J.A. Castellanos and J.D. Tardos. Mobile Robot Localization and Map Building: A Multisensor Fusion Approach. Kluwer Academic Publishers, Boston, MA, 2000.

4. G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. IEEE Transactions of Robotics and Automation, 2001.

5. H. Durrant-Whyte, S. Majumder, S. Thrun, M. de Battista, and S. Scheding. “A Bayesian algorithm for simultaneous localization and map building”. In Proceedings of the 10th International Symposium of Robotics Research (ISRR01), 2001.

6. Sebastian Thrun and Michael Montemerlo. The graph slam algorithm with applications to large-scale mapping of urban structures. Int. J. Rob. Res., 25(5-6):403_429, 2006. ISSN 0278-3649.

7. Sebastian Thrun, Wolfram Burgard, and Dieter Fox. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The MIT Press, September 2005. ISBN 0262201623.

8. Manifold SLAM: a Multi-Agent Simultaneous Localization and Mapping System for the RoboCup Rescue Virtual Robots Competition written by Bayu Slamet Max Pfingsthorn Master in Artificial Intelligence at the Universiteit van Amsterdam.Date of the public defense: December 11, 2006.

9. Andrew Howard, Lynne E. Parker, and Gaurav S. Sukhatme. Experiments with large heterogeneous mobile robot team: Exploration, mapping, dployment and detection. International Journal of Robotics Research, 25(5):431_447, May 2006.

10. StuartRussell, PeterNorvig, ArtificialIntelligence:AModernApproach, Pearson Education, (2002.4).