RoboCup 2021 – Virtual Rescue Robot Firebolts (Iran) Team Description Paper

Nicky Sadighi, Maryam Ahmadian fard, and Ghazal Laghaee

sadighi.nicky@gmail.com
maryamahmadian733@gmail.com
ghazaal.laghaee@gmail.com
http://www.github.com/FireboltsVR

Abstract. This paper is a description of the strategies developed by the *Firebolts* team for participating in the RoboCup 2021 Rescue Simulation League Virtual Robot Competition. The framework used by this team is ROS. Several algorithms are designed for mapping, victim detection, multi-robot exploration, and decision making.

Keywords: RoboCup 2021 Rescue Simulation League Virtual Robot Competition · ROS framework · SLAM · Navigation · Human Recognition · Multi Agent System.

1 Introduction

Firebolts is a team that was first formed in 2018 and has participated in the RoboCup Asia Pacific 2018 competitions, winning first place. This team has improved drastically, starting from a very simple base code. *Firebolts* team members have a lot of experience in National and International RoboCup competitions. Our main motivation for taking part in these competitions is to help improve autonomous exploration and rescue quality so that every day fewer rescuers need to risk their lives by going into ruins after natural or man-made disasters. This year we have mainly focused on efficient victim detection, marking, and also connecting the robots to explore more systematically.

Team members and their contributions:

- Victim Detection: Maryam Ahmadian fard
- Autonomous Exploration: Nicky Sadighi, Ghazal Laghaee
- Quadrotor: Ghazal Laghaee, Maryam Ahmadian fard
- Navigation: Nicky Sadighi, Maryam Ahmadian fard
- User Interface: Ghazal Laghaee, Maryam Ahmadian Fard
- RGBD Slam Multi agent Exploration: Nicky Sadighi

2 System Architecture

Our main software modules are Navigation, Exploration, Localization, Mapping, User Interface, Victim Detection, and Marking. Each robot is provided with these utilities. Our robots are currently not able to communicate with each other directly; although it is our goal to add this option to our code. We use two types of robots. Firstly the Pioneer3AT with the Hokuyo laser scanner and an RGBD camera and odometry sensors, and secondly the quadrotor. Our system is directly implemented in ROS, which is a platform in which tools and information are put to users, and Gazebo, for simulating the urban research as rescue scenarios.

2.1 Mapping

In robotics, the computational problem of building or updating a map of an unknown space while simultaneously keeping track of the position of an agent within this environment is referred to as Simultaneous Localization And Mapping (SLAM) [1]. We are currently using the GMapping algorithm used in the ROS Slam package [2] with enhanced parameters to optimize the map precision. Using the GMapping algorithm, robots can generate a map with the data that their laser gathers from the environment. For our quadrotors, we use Hector Mapping [3]. Robot navigation depends highly on proper mapping because if the robot does not have a map, it may choose a pre-explored target and spend its time merely on a small portion of the map. Furthermore, it might not reach its chosen location due to the wrong path planning.

2.2 Shared Map

Each of our robots generates a separate map of the environment. Creating a shared map is of high priority because by doing so, each robot knows which parts of the map have already been explored; therefore, a huge amount of time and resources can be saved by preventing the robots from exploring parts of the map multiple times. Our team uses the Multi-Robot Map merger package from the ROS repository to publish the map between our robots. The Map Merger ROS node retrieves maps from the Robots and attempts to merge them into a larger map using a modified version of the Iterative Closest Point (ICP) algorithm (Besl and McKay, 1992) [4]. Our output is shown in Fig.1.

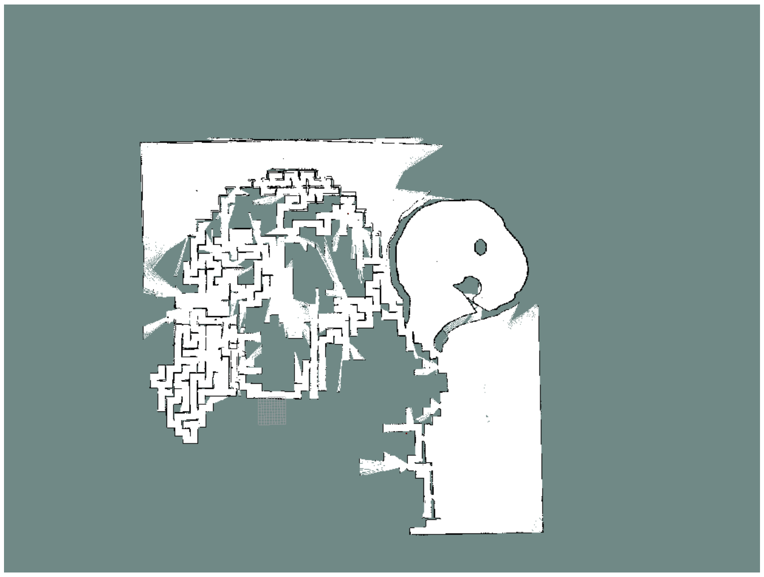

Fig. 1: Firebolts Map from Robocupap 2018 Final Round

2.3 Autonomous Navigation

One of the most important goals of this league is to accomplish a fully autonomous navigation system to become needless of operators. The Navigation Stack takes in information from odometry and sensor streams and outputs velocity commands to send to a mobile base [5]. The main features of our navigation algorithm are a Hybrid obstacle avoidance and Go-to-Goal; and a wall-following path planner node that allows multiple robots to cooperatively build a map. By using an exploration plugin for the navigation module, the robots can autonomously explore their working space and create a shared map for navigation on all robots.

2.4 Victim Detection

Using the RGBD camera, robots receive images of their environment. An algorithm is needed to detect if part of the surrounding is a victim. We developed a code based on the HOG algorithm [6–8]. By giving a database of correct and false samples of humans to the robot, it learns the right shape of victims and compares it to each image frame it receives from its camera. Evidently, the more an environment is crowded and dark, the more challenging it becomes for a robot to accurately recognize victims. We are still working on resolving this issue. The victims we find should be treated differently depending on whether they are still alive or not. There are three ways to understand if a victim is still alive: by their voice, their motion, and their heat; Voice detection is still not used in RoboCup competitions. For motion detection, we have designed an algorithm that makes the robot stop and start comparing two images of the victim. If they are different, it means that the victim is moving and hence alive. Our robots use a thermal camera to detect the heat of alive victims. Our output is shown in Fig. 2.

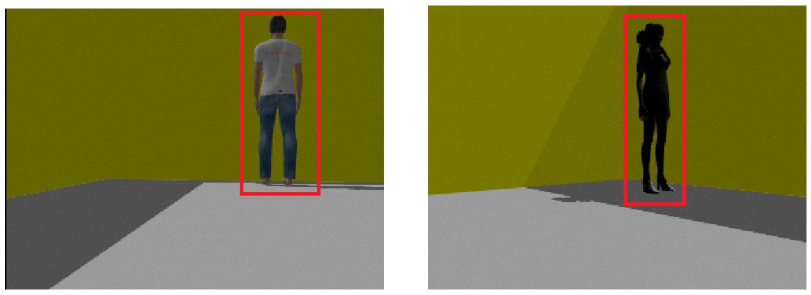
Fig.2: HOG sample data

2.5 Marking

After robots discover a victim, it is important to specify its place in the map they have generated so that rescue teams can find the victim later. For this purpose, we mark the position of the victims on the map using a graphical algorithm. We distinguish dead and alive victims by using red and green stamps for specifying their location, respectively. If the victim is still alive, the robot continues exploring the map and returns to the victim it had found before the end of the competition.

2.6 Controller

We use the *Firebolts* user interface to save the map, start each robot's autonomous exploration, and switch to manual mode whenever desired. We also use a PS4 controller for manually driving the robots in case they are stuck or to separate their paths if they are initially close to each other. Currently we are trying to minimize our use of the controller and rely more and more on the autonomous exploration of our robots.

3 Future Work

There are still many things we aim to improve in our code. These include:

- Leg detection: An algorithm that can detect victims by their legs instead of their whole body. This is extremely challenging because legs can be easily mistaken with objects such as table legs or trees.
- Changing the center of mass of the robots to prevent them from turning over in any situation
- Voice Detection and enhancement of heat detection
- Using 3D mapping
- Improving the navigation of quadrotors

4 Conclusion

In this paper, we have given a short introduction to our system architecture. In the past years, we have enhanced the navigation of separate robots, user interface, and mapping algorithms. This year we are aiming to minimize the exploration time using a shared map for each robot's navigation stack. Also, we are trying to improve our victim detection and marking algorithms.

References

1. `https://en.wikipedia.org/wiki/Simultaneous_localization_and_mapping/`
2. `http://wiki.ros.org/gmapping/`
3. `http://wiki.ros.org/hector_mapping/`
4. Li-Chee-Ming, Julien, and Costas Armenakis. "Feasibility study of using the RoboEarth cloud engine for rapid mapping and tracking with small unmanned aerial systems." The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 40.1 (2014): 219.
5. `http://wiki.ros.org/navigation`
6. N. Dalal, B. Triggs, "Histograms of oriented gradients for human detection.", In Computer Vision and Pattern Recognition, IEEE Computer Society Conference on , Vol: 1, pp. 886 - 893, 2005.

7. D. Navneet and B. Triggs, ”Histograms of oriented gradients for human detection.” In Computer Vision and Pattern Recognition. CVPRIEEE, Vol. 1., 2005.
8. C. Xia and S. F. Sun and P. Chen and H. Luo and F. M. Dong, “Haar-like and HOG fusion basedobject tracking.” In Pacific Rim Conference on Multimedia, pp. 173-182, 2014.