Network Failure Detection and Autonomous Return for PMB-2 Mobile Robot

Dmitry Bereznikov, Aufar Zakiev
Laboratory of Intelligent Robotic Systems (LIRS), Intelligent Robotics Department,
Higher Institute for Information Technology and Intelligent Systems,
Kazan Federal University, 35 Kremlyovskaya street, Kazan, Russia Federation
E-mail: bereznikovdmitry@gmail.com, zaufar@it.kfu.ru
http://kpfu.ru/robolab.html

Abstract

In real world teleoperated tasks a robot connection with its operator is not always stable, so it is important to increase the robot autonomy. This paper focuses on increasing robot autonomy through autonomous return and charging station docking in a case of connection loss. We integrated the algorithm into real robot control system or PAL Robotics PMB-2 robot and experimentally demonstrated its good efficiency. The algorithm analyzes network failure through incoming TCP/IP packets, uses Simultaneous Localization and Mapping (SLAM) and path planning algorithms for autonomous return, and dock station plugin for the robot docking and recharging, which continues until the connection to teleoperator station is restored.

Keywords: mobile robot, algorithm, autonomous return, network failure detection, dock station, PMB-2.

1. Introduction

Mobile robotics is a very practically-oriented field of robotics with robots being widely used in broad range of different tasks[1], including cargo delivery[2], environment mapping and hazardous zones exploration[3,4]. Mobile robots often have operator[5,6], which controls robot actions and monitors robot state remotely from a safe place. Communication between mobile robot and the operator mostly is settled by a wired or wireless connection. Thus, teleoperated robots could lose control because of imperfections in communications technology, human factor or force majeure events[7]. Lost control may lead to entire robot loss or its partial damage[8], which requires mobile robots to demonstrate some level of autonomy in order to avoid such situations.

In this paper, we developed and integrated in PAL Robotics mobile robot PMB-2 control system[9] our network failure detection and autonomous return to dock station algorithm. Our goal was to implement operator-independent autonomous return system that does not require any software changes on operator's side. Therefore, the detection is based on incoming TCP/IP packets analysis. After connection failure is detected, robot returns to its initial position; during the movement, it searches for its dock station in order to start charging until a connection link with the operator restores.

Existing solutions for network failure detection include different approaches to detect connectivity loss. Algorithm[10] requires software configuration on both robot and operator's PC. Papers[11,12] are focused on multiple robots to detect network failure between them. Other methods[13,14] imply special device usage to measure received signal power. The method, which we present in this paper, is developed to overcome these limitations and is experimentally validated.

This paper is organized as follows: Section II describes PMB-2 robot used for experimental validation. Section III is dedicated to network fail detection algorithm. Next, autonomous return is described and experimental results are presented in Section IV. Last Section contains conclusions.

Fig. 1. PMB-2 robot (left) and its charging in the dock station (right); the dock is the white box in the top of the picture.

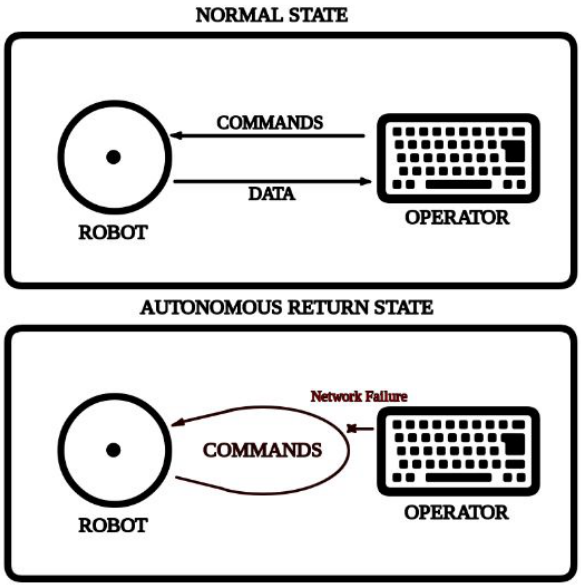

Fig. 2. Network failure detection diagram.

2. System Setup

PMB-2 (Fig. 1) is a PAL Robotics[15] company mobile robot that is equipped with Sick TiM 571 laser range finder (LRF) and 6 DoF inertial measurement units (IMU) sensors, which we use for SLAM and navigation. The robot has 54 cm diameter, 30 cm height, maximal speed of 1 m/s and a round shape, which makes it convenient for indoor use. Its built-in 20,000 mAh Li-Ion battery allows to work autonomously without recharging for a long time. The robot is equipped with 802.11b/g/n/ac Wi-Fi interface and could act as a wireless access point. PAL Robotics also provides charging dock station (Fig. 1, right) with the robot.

PMB-2 runs on Ubuntu 16.04 operating system and ROS Kinetic. Software has preconfigured SLAM and navigation packages. Mapping is performed using *slam_gmapping*[16] ROS package; AMCL[17] is responsible for localization and actual navigation is implemented in *global_planner*[18] package. Operator PC has Ubuntu 16.04 and ROS Kinetic environment installed. However, our system is software-agnostic, thus, it only requires to communicate with robot through TCP/IP connection.

3. Network Failure Detection and Recovery

PMB-2 is controlled through ROS interface and teleoperation mode is also implemented using ROS capabilities. The workflow is following:

(i) PMB-2 turns on internal Wi-Fi interface and creates access point.
(ii) Operator connects to the PMB-2's access point.
(iii) Operator configures ROS on his PC to connect to PMB-2's ROS.

The robot and operator's PC are united into a common network with a full software compatibility, i.e. ROS nodes launched on robot are visible and interactive for ROS nodes launched on operator's PC and vice versa. Such tough integration is very convenient for common software development workflow since ROS nodes are indifferent to source of data or commands. They could come through Wi-Fi, Ethernet, Bluetooth or even Zigbee networks[19] encapsulated in ROS topics. However, this encapsulation hides the data source information, therefore, there is no common way to determine whether topic is disconnected (data could not reach the destination point) or topic is simply empty (data is not sent at all). Therefore, we developed network failure detection node that takes teleoperation commands topic as an input and determines if the connection to operator is alive or not. Connection state is determined by *ping* utility[20], which is built-in into modern OSs as well as into Ubuntu.

At the system startup the node waits for initial teleoperator connection to prevent the robot from returning to a starting point instantly after turning on (Fig. 2). In normal state, ping is not used and no network load by failure detection algorithm is applied. However, if there are no incoming commands for 10 seconds, network failure possibility is recognized by the robot. It automatically turns on the ping utility and checks whether an operator is reachable in the network. Returning ping messages mean that the operator is in the network and does not send commands; if no ping returns, the operator is considered as disconnected and the autonomous return algorithm is launched.

After switching to autonomous return mode, the robot launches a built-in function of an automatic connection to a dock station. This feature allows the robot to safely wait for the operator reconnection if the starting position is successfully reached. Docking feature detects dock station surface laser pattern at up to 1-meter distance and performs a docking maneuver. Successfully docked robot starts to recharge (Fig. 1, right).

During the return movement, the robot continuously checks for operator's availability in the network. If the connection restores, it switches back to teleoperation mode. The autonomous navigation and movement stop; also, if the robot is already at the dock station, it undocks and waits for commands in fully operational state. Thus, no operator assistance is needed for the robot to recover from a network failure.

4. Autonomous Movement

Autonomous movement is fully implemented in ROS framework. When the algorithm detects network failure between robot control system and teleoperator, algorithm sends command to *move_base*[21] node which is part of the implemented at this robot navigation stack. The goal command is filled with position of starting point, orientation of robot and frame ID which is used for global navigation. After sending the goal, *global_planner* node builds a plan on the map from robot current position to goal position. This global plan goes as an input to *move_base*; this node turns the planned path into movement commands to robot's motor drives (Fig. 3). Algorithms performs searching for dock station during the movement to save time for docking.

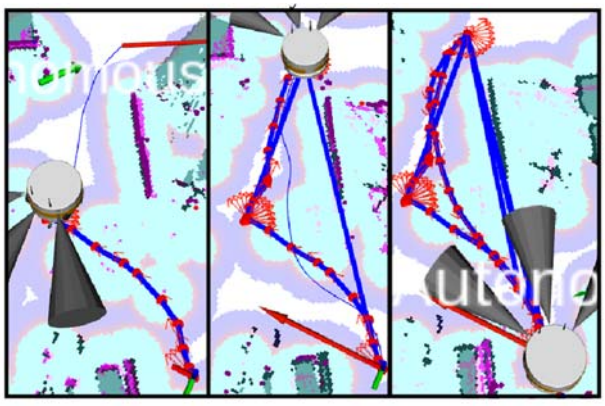

Fig. 3. Network failure detection and autonomous return. Robot is teleoperated (left); network failure detected (center); autonomous return performed (right).

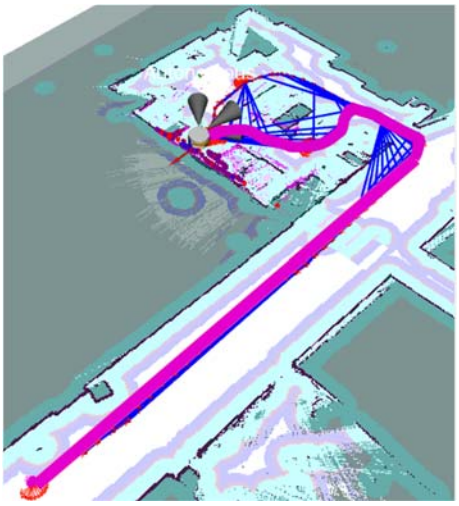

Fig. 4. Travel distance pointed purple line - 24 meters.

Autonomous return process relies on a map that was obtained during teleoperated movement and could be further used for localization and navigation[22]. PMB-2 uses laser-based SLAM *slam_gmapping* node which allows us to construct an occupancy grid map. During autonomous return, the robot is able to localize itself on this map using *amcl* package which uses Adaptive Monte Carlo Localization method.

5. Experiments

We conducted a set of seven experiments to determine the effectiveness and applicability of our algorithm. In each trial, we measured (Table 1):

- the distance traveled by the robot during the autonomous return (second column);
- time intervals from the moment of connection break, to the moment of switching robot to the autonomous return mode (third column);
- return time to the starting point (fourth column).

Experimental results in Table 1 show that delay of network failure detection is not stable and fluctuates around 12-16 seconds. However, it is not dangerous, because if there are no commands from a teleoperator, the robot stops. Figures 3 and 4 illustrate that the robot autonomous return time is more dependent on a path complexity than on a path length. The shorter path had multiple obstacles on the robot's way, which slowed it down. The longer path of the seventh trial consumed relatively little time because of a higher speed on straight segments of the path.

All tests passed successfully; the algorithm detected communication break with the teleoperator, switched to autonomous return and the robot reached starting point, found the dock station and connected for recharge.

Table 1. Algorithm testing results.

Trial number	Traveled distance in meters	Time to detect network failure after disconnect in seconds	Traveled time in seconds	Figures to illustrate
1	4	13	10	Fig. 3
2	4	16	12	
3.	4	14	18	
4	4	13	20	
5	4	12	11	
6	24	15	50	Fig. 4
7	40	15	60	---

6. Conclusions

The paper presents network failure detection algorithm and its usage in autonomous return task. The algorithm is fully compatible with ROS, does not need additional hardware or software installed on operator's PC and could be generalized for any TCP/IP connections. Both network failure detection method and autonomous return were experimentally validated on PMB-2 robot and showed their practical applicability for path planning[23].

Acknowledgements

This work was supported by the Russian Foundation for Basic Research (RFBR), project ID 19-58-70002.

References

1. K. Shabalina, et al., Comparative Analysis of Mobile Robot Wheels Design, in *Int. Conf. on Developments in eSystems Engineering* (IEEE, 2018), pp. 175-179.
2. I. Shimchik, et al., Golf cart prototype development and navigation simulation using ROS and Gazebo, in *MATEC Web of Conf.* **75** (EDP Sciences, 2016), p. 09005.
3. E. Magid, R. Lavrenov and I. Afanasyev, Voronoi-based trajectory optimization for UGV path planning, in *Int. Conf. on Mechanical, System and Control Engineering* (IEEE, 2017), pp. 383-387.
4. M. Sokolov, et al., Modelling a crawler-type UGV for urban search and rescue in Gazebo environment, in *Int. Conf. on Artificial Life and Robotics* (2017), pp. 360-362.
5. I. Farkhatdinov, JH. Ryu and J. Poduraev, A user study of command strategies for mobile robot teleoperation, in *Intel Serv Robotics* **2** (2009), pp. 95-104.
6. M. Wang and James N.K. Liu, Interactive control for Internet-based mobile robot teleoperation, in *Robotics and Autonomous Systems* **52**(2-3) (2005), pp. 160-179.
7. E. Magid, et al., Artificial Intelligence Based Framework for Robotic Search and Rescue Operations Conducted Jointly by International Teams, in *Int. Conf. on Electromechanics and Robotics "Zavalishin's Readings"* (Springer, 2019), pp. 15-26.
8. K. Nagatani, et al., Emergency response to the nuclear accident at the Fukushima Daiichi Nuclear Power Plants using mobile rescue robots, in *J. of Field Robotics* **30**(1) (2013), pp. 44-63.
9. P. Jordi, L. Marchionni and F. Ferro, Tiago: the modular robot that adapts to different research needs, in *Int. workshop on robot modularity IEEE IROS* (2016).
10. N. Alishev, et al., Network Failure Detection and Autonomous Return Algorithms for a Crawler Mobile Robot Navigation, in *Int. Conf. on Developments in eSystems Engineering* (IEEE, 2018), pp. 169-174.
11. A. Derbakova, N. Correll and D. Rus, Decentralized self-repair to maintain connectivity and coverage in networked multi-robot systems, in *Int. Conf. Robotics and Automation* (IEEE, 2011), pp. 3863-3868.
12. M. A. Hsieh, et al., Maintaining network connectivity and performance in robot teams, in *J. of field robotics* **25**(1-2) (2008), pp. 111-131.
13. D. Tardioli, et al., Enforcing network connectivity in robot team missions, in *Int. J. of Robotics Research* **29**(4) (2010) pp. 460-480.
14. A. H. Mong-ying, et al., Towards the deployment of a mobile robot network with end-to-end performance guarantees, in *Robotics and Automation* (IEEE, 2006), pp. 2085-2090.
15. PAL Robotics, official site: http://pal-robotics.com/
16. M. Quigley, et al., Sub-meter indoor localization in unmodified environments with inexpensive sensors, in *IROS* (IEEE, 2010), pp. 2039-2046.
17. D. Fox, Adapting the sample size in particle filters through KLD-sampling, in *Int. J. of robotics research* **22** (2003), pp. 985-1003.
18. global_planner, ROS package: http://wiki.ros.org/global_planner
19. A. Fernandes, Ad hoc communication in teams of mobile robots using zigbee technology, in *Computer Applications in Engineering Education* **23.5** (2015), pp. 733-745.
20. Ping utility, Ubuntu xenial package: https://packages.ubuntu.com/xenial/iputils-ping.
21. move_base, ROS package: http://wiki.ros.org/move_base
22. A. Zakiev, et al., Path planning for Indoor Partially Unknown Environment Exploration and Mapping, in *Int. Conf. on Artificial Life and Robotics* (2018), p.399-402.
23. E. Magid, et al., Combining Voronoi graph and spline-based approaches for a mobile robot path planning, in: *Informatics in Control, Automation and Robotics ICINCO 2017, Lecture Notes in Electrical Engineering, **495*** (Springer, Cham, 2020), pp. 475-496.