PROBLEMY MECHATRONIKI
UZBROJENIE, LOTNICTWO, INŻYNIERIA BEZPIECZEŃSTWA

ISSN 2081-5891

8, 4 (30), 2017, 37-46

PROBLEMS OF MECHATRONICS
ARMAMENT, AVIATION, SAFETY ENGINEERING

Multi-robot Control via Smart Phone and Navigation in Robot Operating System

Khadir BESSEGHIEUR, Wojciech KACZMAREK*, Jarosław PANASIUK

Military University of Technology, Faculty of Mechatronics and Aerospace
2 gen. Witolda Urbanowicza Street, 00-908 Warsaw
**Corresponding author's e-mail addres: wojciech.kaczmarek@wat.edu.pl*

Received by the editorial staff on 10 April 2017.
Reviewed and verified version received on 29 August 2017.

DOI 10.5604/01.3001.0010.7316

Abstract. Robot Operating System (ROS) is an open source robot software framework which provides several libraries and tools to easily conduct different robot applications like autonomous navigation and robot teleoperation. Most of the available packages across the ROS community are addressed for controlling a single robot. In this paper, we aim to extend some packages so, they can be used in multi-robot applications on ROS. Mainly, the multi-robot autonomous navigation and multi-robot smart phone teleoperation are addressed in this work. After being extended and compiled, the new packages are assessed in some simulations and experiments with real robots.
Keywords: mobile robots, Robot Operating System, multi robots SLAM, multi-robot teleoperation

This paper is based on the work presented at the 21th International Workshop on *CAD, CAM and CAE Systems*, Jurata, Poland, May 8-12, 2017.

1. INTRODUCTION

Multi-robot systems present a more robust and cheaper solution to certain tasks that are better performed using several low-cost robots rather than single, complex ones. Instead of building a powerful single vehicle, a group of vehicles provides the flexibility in performing the task required, as well as makes the system more tolerant to possible individual vehicle faults. The ability to control vehicles as well as to allow them to work cooperatively together has long been a challenging idea in robotics and artificial intelligence.

The multi robot applications vary from industrial applications such as goods transportation, distributed assembly, and infrastructure inspection to military applications such as planetary exploration, assembly in space and urban search and rescue. Among several multi-robot applications, navigating and teleoperating multi-robot systems are considered in this paper. Recently, a lot of research has been conducted in this area. In [1], a new Simultaneous Localization and Mapping approach applicable to multiple mobile robots is described where the 2D Laser sensors data is used to build a dynamic representation of the environment. Some mobile robots building architectures are proposed in [2], where a smart phone is considered as the robot control unit. It is worth noting that the mobile robot is based on ROS, which is the same case in our study.

Robot Operating System 'ROS' [3] is an open source robot software framework which provides several libraries and tools helping the researches to conduct their experiments on the mobile robots carrying ROS. Most of the available packages across the ROS community are addressed for controlling a single robot. In this paper, we aim to extend some packages, so they can be used in multi-robot applications on ROS. We developed new packages that allow the simulation of multi-robot autonomous navigation. The second contribution is a smartphone application which a multi-robot system can be controlled through.

The next section of this paper is dedicated for introducing ROS and its concepts as well as our real robots 'Turtlebot' which the experiments are carried on. The autonomous navigation package for controlling one robot under ROS and how it could be upgraded to the multi-robot case is described in section three. Fourth section discusses the mobile phone robot teleoperation package for one robot, then for several robots. Finally, we conclude this paper in the last section and we draw a set of some future work directions.

2. BACKGROUND

2.1. Robot Operating System

ROS is the Robot Operating System framework, which is used nowadays by hundreds of research groups and companies in the robotics industry. ROS is an open-source, meta-operating system for the robot.

It provides the services expected from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management. It also provides tools and libraries for obtaining, building, writing, and running code across multiple computers. It is based on the concepts of nodes, topics, messages, and services. Here is a brief description of the ROS concepts.

A node is an executable program that performs computation. Nodes communicate with each other by passing messages. A message is strictly a typed data structure. Standard primitive types (integer, floating point, Boolean, etc.) are supported, as are arrays of primitive types and constants. ROS provides an easy way for passing messages and establishing a communication between nodes, which are running independently and communicate with each other over Topics. A topic is a simple string, for instance we have used the topic 'odom' for publishing the robot's positions and velocities. To get these data, a node needs to subscribe to the corresponding topic [6].

However, topics are asynchronous, synchronous communication is provided by services. Services act in a call-response manner where one node requests that another node execute a one-time computation and provides a response. Moreover, ROS provides a set of powerful tools to inspect the state of the system, which include the node's graph, with all the connections (publishers and subscribers) among topics. For more details about ROS, the reader can refer to [4].

2.2. Turtlebot

Turtlebot is a highly capable autonomous mobile platform for developing robot application. The hardware structure can be divided into three main parts: A Kobuki base, a netbook computer, and sensors. The Kobuki base is a differential drive mobile robot base with two passive caster wheels for balancing. Our netbook runs Ubuntu 14.04 LTS OS and has ROS Indigo running on it which takes care of the communication between different sensors and actuators.

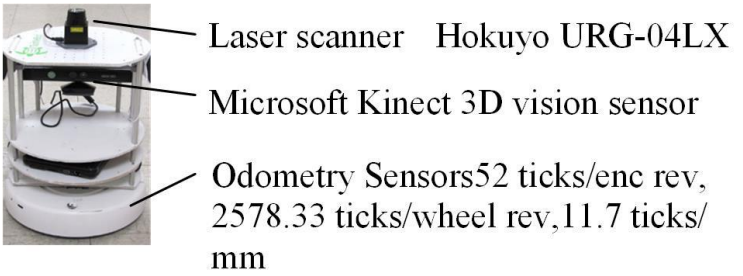

Fig. 1. TURTLEBOT description [7]

Specifications of all the sensors present in the mobile robot are provided in Fig. 1. Kinect is a 3D vision sensor that provides RGBD data, i.e. depth information of every point in the image along with the colour image of the scene. The colour images acquired from Kinect are displayed in an interface as a video feed and the depth data is used to display distance of the center point in the image. This enables the operator to understand the distances of objects in the scene. Besides Kinect, Turtlebot holds an odometry sensor, which provides the position of the robot.

3. AUTONOMOUS NAVIGATION UNDER ROS

In this section we describe how a mobile robot can autonomously navigate within an indoor environment using ROS packages. Then, the procedure is extended to be used for a group of robots' autonomous navigation. ROS provides useful and easy-to-use packages for mapping an indoor environment and autonomously navigating a mobile robot.

After setting-up the communication between the workstation and the robot netbook, a 2D map of the environment is built with the help of the 'GMAPPING' ROS package. During a map generation process, the robot is 'teleoperated' around from the workstation; the system converts the point cloud from Kinect and the odometers data from the encoders to build the 2D map. By running RVIZ, which is a 3D visualizer for displaying sensor data and state information from ROS (see Fig. 2), we can decide when the generated map is satisfying and then save it on the robot netbook using the 'MAP_SERVER' ROS package. Now, the mobile robot can autonomously navigate with the help of the navigation stack which takes a goal as input in the form of geometry msgs/PoseStamped message from the Rviz interface run on the workstation with the button 2D nav/goal (see Fig. 2).

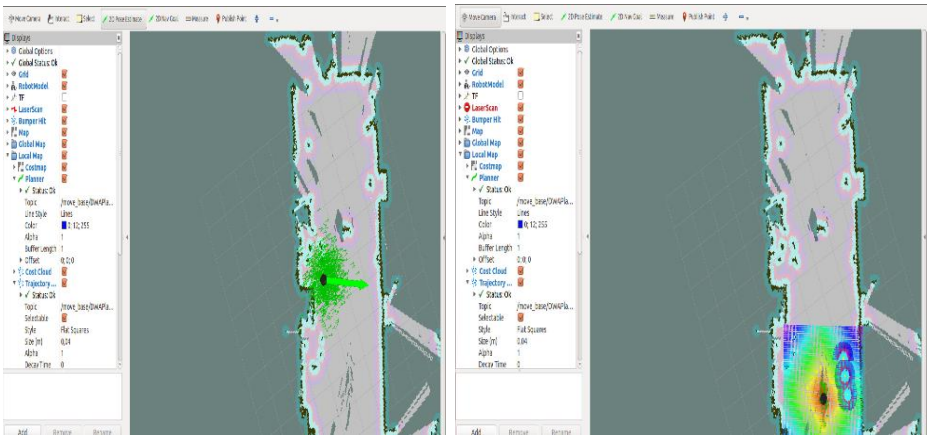

Fig. 2. RVIZ visualization of a single mobile robot autonomous navigation process

This message provides the coordinates of the final goal along with orientation. Global and local planners plan the trajectory to reach the goal and it is sent to the robot through a predefined topic.

The robot successfully navigates to the goal point with avoiding obstacles, even dynamic ones. Regarding how useful this package is, it is worth extending it to autonomously navigate several robots since the RVIZ interface buttons can communicate with only one robot. Our idea is to modify the RVIZ interface, so that different mobile robots can be addressed from the same interface

In the multi-robot case and where the RVIZ simulator is needed for testing some multi-robot SLAM or exploration algorithms for instance, the explained procedure of navigating a single robot is extended to work for the multi-robot case. For this, a virtual multi-robot system is considered in the simulation tool GAZEBO. The simulation environment consists of two robots and some static obstacles (see Fig. 3) where the robots are required to autonomously navigate with avoiding obstacles to their corresponding goal points.

First, the necessary launch files for setting up the environment and executing the navigation algorithms are all included in the main launch. The latter consists of three parts: setting up the simulation environment on GAZEBO, executing the necessary nodes' programs for navigating the first robot, then for the second robot. Each of the last two parts contains the required launch files for spawning a virtual robot, executing the map server, and launching the SLAM algorithm. They are identical but each one is launched under a specific namespace for each robot. In addition, the 'tf_prefix' parameter has to be unique for each robot. Secondly, the visualization tool RVIZ is adapted, so that both of the two robots can be controlled at the same time. The RVIZ package is modified and compiled. The new RVIZ interface holds two buttons for sending the corresponding goal points to each robot. In addition, the 'tf_prefix' parameter is included, so that each robot is considered separately.

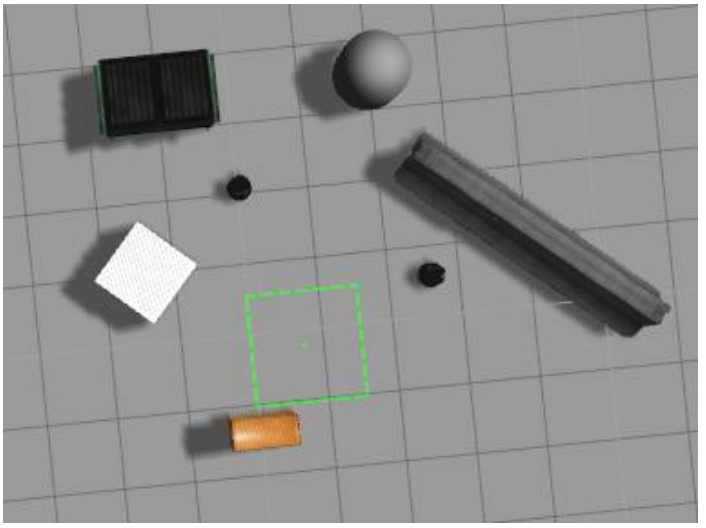

Fig. 3. The simulation environment of a multi-robot system in Gazebo

A simulation is carried out on GAZEBO using the new launch files and RVIZ package [8]. The results are illustrated in Fig. 4. Clearly, both robots simultaneously and safely navigate to their corresponding goal points which is illustrated in the same RVIZ interface. This new framework is convenient for testing new multi-robot environment exploration or multi-robot SLAM algorithms.

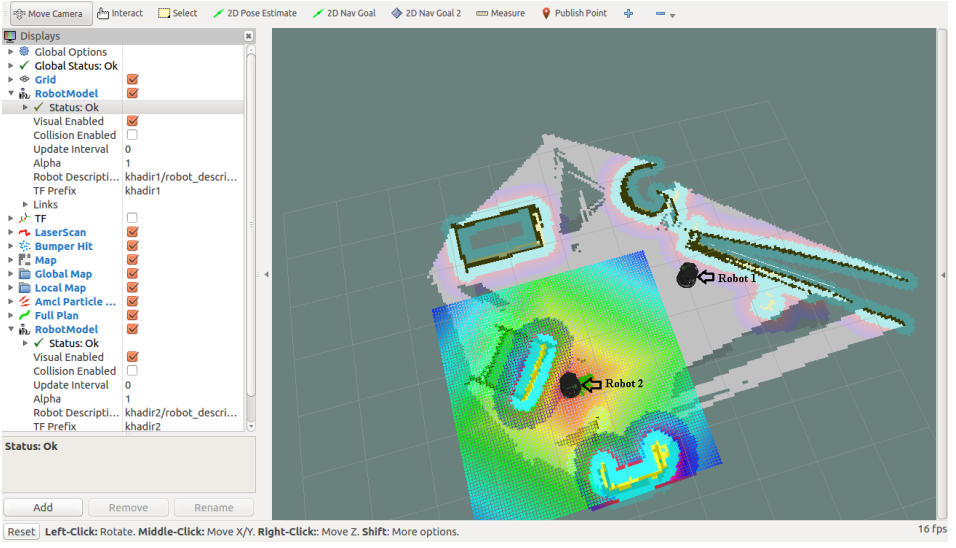

Fig. 4a. Multi-robot navigation in RVIZ (initial positions)

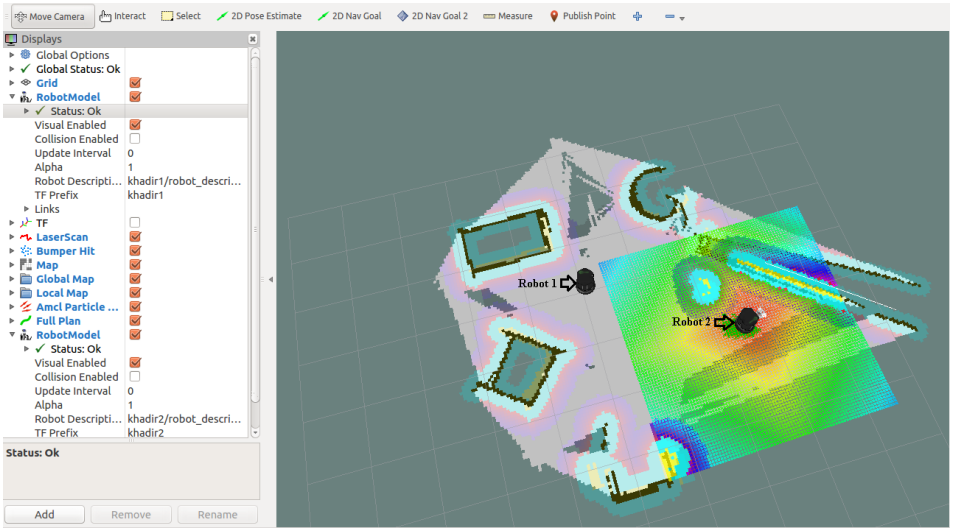

Fig. 4b. Multi-robot navigation in RVIZ (final positions)

4. ANDROID APPLICATION FOR TELEOPERATING MULTIPLE ROBOTS

An application named ROS Teleop [5] has been developed and used by thousands of users. This application enables users to teleoperate a ROS-enabled robot using an Android device. Once the mobile phone is connected to the robot's netbook via wireless communication, the robot can be driven around using the joystick displayed on the mobile phone screen.

In addition to the joystick, the application allows a feedback from the robot as the user receives the images captured by the robot's camera. However, the application is designed to teleoperate only one robot at a time. Our aim is to extend the use of this app for teleoperating several robots at the same time.

Android applications that extend a functionality of devices are developed primarily in Java programming language using the Android software development kit (SDK). However, in order to develop our app, the knowledge on developing codes is not limited to the Android application development and how to develop codes in Android Studio only, but the interrelationship between Android and ROS needs to be considered too as the Android application has to be compatible with the enabled-ROS robot. The link between the Android app and the ROS environment is ensured by Rosjava which is a java implementation of ROS that includes both the roscore and nodes. Rosjava Android is the extension to rosjava for use with Android. It allows java applications to interface with ROS topics, services, and parameters.

Using both of ROS, Arduino Studio and Rosjava, an android application is developed that allows users to control multiple robots and visualize feedbacks from the robots. The idea is to develop a similar interface like in the teleop application (with the joystick and the visualization parts) and add new buttons and a text zone which allows the user to select the desired robot to control among the other robots.

In the application interface (see Fig. 5), a text zone is used to specify the robot's number that we want to communicate with. Furthermore, a "begin" button is created to start sending the joystick commands and receiving the images from the designated robot. In case a second robot control is desired, the "Stop" button allows ending the communication with the first one, then the number of the next desired robot to control should be entered.

An experiment is conducted using the two robots, a personal computer, and a smart phone. First, a communication network is set up which consists of two netbook laptops, the smart phone and the workstation pc. The latter is used as the main master of the network. On each robot's netbook, the necessary launch files for uploading the robot and the camera are executed under a specific namespace and with a unique 'tf_prefix' parameter for each robot.

Both of the robots can now be teleoperated from the smart phone after launching the developed app on the latter. The number of the desired robot to control must be entered in the specified text zone.

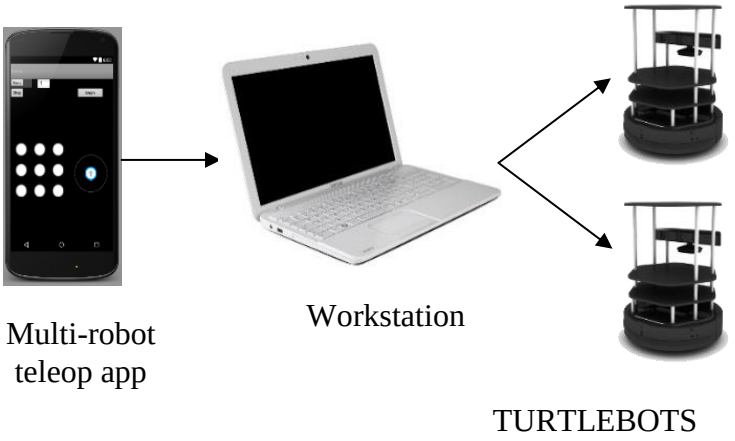

Fig. 5. The framework architecture

The developed application has been successfully tested using a personal smart phone and two real mobile robots 'TURTLEBOTS' where the two robots respond to the joystick commands and the images are displayed on the phone screen. In addition, we can switch dynamically from controlling one robot to another.

5. CONCLUSIONS

In this work, two tasks for the ROS-enabled mobile robots are addressed: autonomous navigation and smart phone teleoperating. Our aim is to extend some single robot ROS packages, so they can be used in the multi-robot case. New launch files, which include the modified and recompiled navigation packages, makes it possible to autonomously navigate several robots through the same RVIZ interface. To teleoperate a group of robots using a smart phone, an Android app that allows us to select the desired robot to control is developed. The app interface consists of a joystick, which sends the control commands to the selected robot through a workstation, and a visualization part that displays the current image being captured by the selected robot's camera. The new codes are compiled and successfully tested through a simulation in GAZEBO and experiments using two real robots 'Turtlebot'. Our future works lie in addressing the implementation of the formation control algorithms on ROS-enabled mobile robots.

REFERENCES

[1] Koch Philipp, Stefan May, Michael Schmidpeter, Markus Kühn, Christian Pfitzner, Christian Merkl, Rainer Koch, Martin Fees, Jon Martin, Daniel Ammon, Andreas Nüchter. 2016. "Multirobot localization and mapping based on signed distance functions". *Journal of Intelligent & Robotic Systems* 83 (3-4) : 409-428.

[2] Aroca V. Rafael, Antônio Péricles, B.S. de Oliveira, Luiz Marcos, G. Gonçalves. 2012. Towards smarter robots with smartphones. In *Proc. 5th Workshop on Applied Robotics and Automation* 1-6.

[3] Quigley Morgan, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, Andrew Ng. 2009. ROS: an open-source Robot Operating System. In *Proceedings of ICRA workshop on open source software* 3 (3.2): 5.

[4] Quigley Morgan, Brian Gerkey, William D. Smart. 2015. *Programming Robots with ROS, a Practical Introduction to the Robot Operating System*. First. O'Reilly Media, Inc.

[5] https://play.google.com/store/apps/details?id=com.github.rosjava.android_apps.teleop.indigo (access 10 April 2016).

[6] www.ros.org (access 10 April 2016).

[7] Besseghieur Khadir, Wojciech Kaczmarek, Jarosław Panasiuk, Piotr Prusaczyk. 2017. Formation control of mobile robots under ROS. In *Proceedings of 23rd International Conference on Engineering Mechanics* 138-141.

[8] Baranowski Leszek, Wojciech Kaczmarek, Jarosław Panasiuk, Piotr Prusaczyk, Khadir Besseghieur. 2017. Integration of vision system and robotic arm under ROS. In *Proceedings of 23rd International Conference on Engineering Mechanics* 114-117.

Sterowanie wieloma robotami za pomocą smartfona i nawigacja robotów w otwartym systemie operacyjnym ROS

Khadir BESSEGHIEUR, Wojciech KACZMAREK, Jarosław PANASIUK

Wojskowa Akademia Techniczna, Wydział Mechatroniki i Lotnictwa, ul. gen. W. Urbanowicza 2, 00-908 Warszawa

Streszczenie. Otwarty system operacyjny ROS (Robot Operating System) udostępnia wiele bibliotek i narzędzi wspierających tworzenie aplikacji dla robotów (np.: autonomiczna nawigacja i telemetria). Przy czym większość opracowanych pakietów ROS umożliwia jedynie kontrolowanie pojedynczego robota. W artykule przedstawiono możliwości rozszerzenia wybranych pakietów, tak aby mogły być używane w aplikacjach wielorobotowych. Poruszono temat wielozadaniowej autonomicznej nawigacji i wielozadaniowej inteligentnej teleoperacji. W ramach prowadzonych prac rozszerzono możliwości pakietów, a ich funkcjonalność przedstawiono na przykładzie wybranych symulacji oraz testów przeprowadzonych z użyciem rzeczywistych robotów.

Słowa kluczowe: roboty mobilne, ROS – Robot Operating System, symultaniczna lokalizacja i mapowanie SLAM, teleoperacja wielu robotów