Hands-on Robotics Teaching with ROS

Roel Pieters, Reza Ghabcheloo and Minna Lanz

Tampere University of Technology, Tampere, Finland,
roel.pieters@tut.fi

Abstract. This paper describes the teaching efforts in several courses in our newly started robotics major in Tampere University of technology. While the fundamentals and theory in robotics are covered by traditional courses, hands-on experiments and student projects use ROS as software platform. ROS is introduced by an overview lecture with practical demonstration and aims to point students towards information and resources instead of providing ready-made solutions. This approach leads to much trial-and-error and a steep learning curve that, we believe, is highly valuable. We reflect on an early evaluation of our approach and conclude that the advantage of ROS as tool to educate robotics is due to its holistic nature. From low-level concepts (e.g. interfacing, communication, diagnostics) to high-level functionality (e.g. visualization, manipulation, SLAM), all are included in ROS and required to educate the next generation roboticists.

Keywords: robotics, education, Robot Operating System

1 Introduction

Since the advent of robotics, considerable effort has been put in the development of theory, algorithms and their implementation. Similarly, such effort should also be represented in the education of robotics at academic institutes [1–4]. The balance between theoretical foundations and practical work is, however, not trivial to find, and often complicated by the multitude of available software. Traditionally, robotics education has gone hand-in-hand with programming (e.g. C/C++) or higher-level computing environments such as Matlab®. In recent years, however, research has moved slowly to adopt ROS as middleware [5], and many other libraries and interfaces offer some form of integration towards it (e.g. ROS in Matlab). Education has caught up to this and many higher institutes of education now offer ROS in course form, or require ROS to be used as programming interface for project work[1]. Additionally, professional education on ROS that is not part of a curriculum can be found in summer schools [6], conferences[2], online courses and in bookform[3]. The take-up of industry, however,

[1] https://wiki.ros.org/Courses
[2] https://roscon.ros.org
[3] https://wiki.ros.org/Books

is slower but initiatives are ongoing to provide the reliability and robustness that is required in industrial environments (EU H2020 project ROSIN[4] [7]).

As educational design is mostly done behind closed doors, with this brief paper we report the route taken at Tampere University of Technology, in the recently started robotics major. By providing an overview of our approach and an early evaluation of outcomes, we hope to open up a discussion on best practices with respect to robotics education and ROS.

2 Robotics major

The robotics major at Tampere University of Technology started in September 2017, and offers MSc students to specialize in robotics and the technology related to it. The degree is offered by the Faculty of Engineering Sciences. The most significant contributions come from the laboraties of Automation and Hydraulic Engineering (AUT) and Mechanical Engineering and Industrial Systems (MEI), although other units of the university, for example Signal Processing, also contribute. The two year program (120 ECTS) requires students to have solid mathematical background and good programming skills. Studies include the fundamentals of robotics, control of robotics, sensing/perception systems for robotics, robot programming and project work. After the studies, students are able to develop robot-based systems. The major consists of four mandatory courses complemented with elective courses that go deeper in a certain topic. In several of these courses, experimental work includes the use of ROS to connect to and control robotic systems, as described in Table 1 and 2.

2.1 TUTRobolab

While all traditional lectures are given in lecture rooms, practical work and demonstrations make use of a laboratory environment specifically designed for students. This TUTRobolab offers a place for students to work with robotic equipment and experiment without major restrictions. Robots and related technology are for example industrial manipulators (Universal robots, Franka, PRob), mobile robots (turtlebot, in-house developed robot), a multitude of sensors (2D/3D ToF cameras, lidar, GPS, IMU, etc.) and different processing platforms (PCs, embedded PCs, Raspberry Pi). While giving preference to course students, the lab is available 24/7 to all students interested in robotics and aims to create a casual learning environment.

2.2 Mandatory courses

The core of the major consists of four mandatory courses, as described in Table 1, designed to educate students on the basics and practicals of robotics and includes project-based work to prepare them for future professional work life. All courses emphasize practical implementation and require programming to be part of the work.

[4] http://rosin-project.eu/

Table 1: Mandatory MSc courses as part of the robotics major

Mechatronics and Robot Programming	*5 ECTS*	*Lab exercises*
Provides an introduction to sensors and robots. Student groups (2-4 persons) complete exercises ranging from sensor interfacing and read-out, visualization, installation and set-up for different processing platforms (PC, Raspberry Pi, wireless networking, SSH) and robot platforms (motors, turtlebot, crane), to advanced algorithms (simulation, SLAM, navigation). All exercises are demonstrated by each group, questioned to assess each member's understanding and finalized by a report.		
Robotics Project work	*6 ECTS*	*Group project*
Provides project-based group work. Student groups (2-4 persons) pick a topic and develop a robotics project, emphasized on interdisciplinary collaboration. Lectures are provided by speakers from industry and academia. The project requires students to develop project planning and management skills, and collaborate on multiple disciplines. Typical projects include LEGO house building with a UR5 robot, indoor delivery robot and upgrading an industrial robot manipulator.		
Robot Manipulators: Modeling, Control and Programming	*4 ECTS*	*Theory, practicals*
Provides a traditional introduction to robot manipulation. In this course students obtain knowledge on robot modeling (kinematics and dynamics, Denavit-Hartenberg notations), and programming for industrial manipulators. Exercises use Matlab and the course is finalized with an exam.		
Fundamentals of Mobile Robots	*5 ECTS*	*Theory, practicals*
Provides the fundamentals of mobile robotics including localization, mapping and planning. Extensive exercises use Matlab and ROS.		

2.3 Elective courses

Besides the four mandatory courses, additional credits have to be taken to complete the major. These courses can be on specific topics such as computer vision and machine learning, or more focused on specific fields within robotics, such as planning and advanced control (e.g. see Table 2). In both mandatory and elective courses, where possible, ROS is used to ease the education of robotics and robotics related technology.

2.4 Teaching philosophy

The approach we take to educate robotics on a MSc level, is to find an appropriate balance between theory and practice. While the fundamentals can only be taught by introducing the relevant mathematical theory, a deeper understanding is obtained when such theory can be tried and tested in simulation or experiments. Take for example coordinate systems and their transformation.

Table 2: Elective MSc course as part of the robotics major that includes ROS

Advanced Robotics	*5 ECTS*	*Theory, practicals*
Provides advanced theory on motion control for robots, such as dynamical systems, stability theory, visual servoing, force control, etc. Theory is supported by practical implementation and integration on an advanced robot manipulator (Panda, Franka Emika GmbH).		

Understanding the theory and its use can be assisted by tools such as Matlab[5] and ROS[6] that ease implementation and visualization.

A choice was made to not have a separate course purely on ROS, as other institutes offer (such as e.g. ETH Zürich[7]), but only provide an introductory lecture. This 1.5 hour lecture provides the basics of Linux and ROS, and shows students where to find information and how to develop their skills. In case of difficulties or problems student assistants are available to assist and help out with practical matters.

3 Evaluation

The first intake of the major attracted a significant number of students, where a selection had to be made based on background and motivation. Such decisions are difficult as grades and background do not always represent the best selection criteria. For example, giving preference to students that have experience and background with ROS might be suitable for elective courses (e.g. Advanced Robotics), but not for mandatory courses, as ROS will be introduced there. Despite this, the majority of students successfully passed, or are about to pass, the courses. Evaluation of (ROS) skills was assessed by oral examination and practical demonstration of the developed software.

3.1 Lessons learned

Lessons learned for the educators of the robotics major are mostly related to the time and effort necessary to prepare course work and practical experiments. It is very easy to underestimate the time required to set up simulations and experiments and verify that all will function properly on different platforms. This applies similarly for laboratory systems that are in continuous use for different projects. Software versions and hardware differences might simply not be compatible. It is difficult to anticipate for this and to estimate how much time and effort it takes for students to master such realization.

[5] https://se.mathworks.com/help/robotics/gs/coordinate-systems-in-robotics.html
[6] https://wiki.ros.org/tf2
[7] http://www.rsl.ethz.ch/education-students/lectures/ros.html

Even though ROS is a major step forward in providing a standard in robotics education, this also is a drawback. It is very convenient to install a package, execute the algorithms it provides and demonstrate a functioning system. This does not imply, however, that the fundamentals behind the algorithms and robot are understood. Such understanding has to come from studying the algorithms and experimenting with different parameters, or by developing and/or extending the methods. Again, we believe that by providing the tools and problems, and requiring from students a solution, will lead to a deeper understanding of the theory.

Additionally, not all students have the same background, leading to gaps in programming capabilities and hands-on experiments with software coding. This leads to big differences in outcomes and progress throughout a course. One solution to this is to require a certain level of programming skills in order to be accepted in the major. Despite this, we found good results in creating group work with different skill sets. Even though this might cause one person to take the lead in project work, this can be compensated to ensure all participants have to demonstrate the learned skills. Group work also leads to students learning valuable skills necessary in professional work life, such as educating colleagues, team work, effctive time and project management, and communication.

Feedback that was obtained from students after finalizing a course gave valuable information on how to improve the education. Issues that were raised were for example, the lack of a lab technician for quick questions and small practical issues, and the deployment of a virtual workspace where students can collaborate, discuss and share ideas (e.g., Slack[8]).

3.2 Post-graduate requirements

In Finland, it is common to do a MSc thesis in a company, where ROS is most likely not being used. Why should an academic institute then spend so much effort on educating their students with ROS? Our answer to this is that it's impossible to ensure a common framework that is supported by all involved partners. Even the most common tool for studying engineering (Matlab), is still underrepresented in industry due to its high cost. Our main aim is to educate future engineers with the skills and capabilities to design robotic systems. This whole overview of robotics ranges from interfacing sensors and actuators, to control architectures and high-level industrially relevant applications. With so much diversity a common ground is found in ROS, as it supports all. Moreover, different programming languages can be used, to accomodate different educational backgrounds. Through ROS, students practice C/C++ or Python which is commonly used in industry, and learn what a complex robot control system should include.

[8] https://slack.com/

4 Conclusion

This short paper described the efforts made in Tampere University of Technology with respect to robotics education. Our approach is to balance theory with significant hands-on experimental work, by utilizing the Robot Operating System. The advantage that ROS has over other robotics software, such as Matlab, is that it provides a complete overview and implementation of required tools. From low-level concepts such as communication and interfacing to high-level algorithms, it educates roboticists the basics and pushes the state-of-the-art. Drawbacks of ROS are found in the limited adoption in industry and in compatibility or support issues (e.g., unsupported versions, hardware/software version mismatch). Despite this, a first evaluation of our robotics major suggests that students benefit greatly from a community-supported initiative such as ROS. The large user-base and growing support for robotic equipment, means that information can be found and asked for on-line. This brings a steep learning curve that, we believe, is highly valuable.

Acknowledgements

The work presented is financed by the Academy of Finland project: "Competitive funding to strengthen university research profiles", decision number 310325.

References

1. Touretzky, D.S.: Computer Science Education: Robotics for computer scientists: whats the big idea? Computer Science Education, vol. 23, no.4, pp. 349-367, 2013.
2. Correll, N., Wing, R. and Coleman, D.: A One-Year Introductory Robotics Curriculum for Computer Science Upperclassmen. IEEE Transactions on Education, vol. 56, no. 1, pp. 54-60, 2013.
3. Esposito, J.M.: The State of Robotics Education: Proposed Goals for Positively Transforming Robotics Education at Postsecondary Institutions. IEEE Robotics & Automation Magazine, vol. 24, no. 3, pp. 157-164, 2017.
4. Sergeyev, A., Alaraje, N., Parmar, S., Kuhl, S., Druschke, V. and Hooker, J.: Promoting industrial robotics education by curriculum, robotic simulation software, and advanced robotic workcell development and implementation. Annual IEEE International Systems Conference (SysCon), Montreal, QC, pp. 1-8, 2017.
5. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R. and Ng, A.Y.: ROS: an open-source Robot Operating System. In ICRA workshop on open source software, Vol. 3, No. 3.2, pp. 5, 2009.
6. Ferrein, A., Kallweit, S., Scholl, I. and Reichert, W.: Learning to program mobile robots in the ROS summer school series. In Proceedings of the 6th International Conference on Robotics in Education (RIE-15), 2015.
7. Ferrein, A., Schiffer, S. and Kallweit, S.: The ROSIN Education Concept. In Iberian Robotics conference, Springer, Cham, pp. 370-381, 2017.