From simulation to a real system—A ROS-Based Multi-Robot System

Bachelor's Thesis

Supervisor: Prof. Dr. Alexander Pretschner
Advisor: Ana Petrovska
Email: {alexander.pretschner, ana.petrovska}@tum.de
Phone: +49 (89) 289 - 17830
Starting date: immediately

Fakultät für Informatik
Lehrstuhl 4
Software & Systems Engineering
Prof. Dr. Alexander Pretschner

Boltzmannstraße 3
85748 Garching bei München

Tel: +49 (89) 289 - 17830
https://www4.in.tum.de

Context

Modern CPSs need to be able to operate in complex, constantly changing, uncertain and unanticipated environments, and yet remain efficient and reliable. They find their application across different domains: energy, manufacturing, environment protection, medicine, traffic control, and currently most prominently in the areas of autonomous vehicles and robotics. Self-adaptation is widely considered as one of the key approaches to deal with the challenging problem of change and dynamicity in these systems. However, as pointed out across different studies, the validation of research contributions in this field is often limited to simple example applications [1, 2, 3]. A current challenge that still remains open is to develop robust approaches for self-adaptivity and demonstrate their applicability and value in practice on real systems, through controlled experiments and case studies [4].

Goal

The goal of this thesis is two-fold. First, assembling and setting up the hardware of two mobile ground robots TurtleBot3 Burger [5, 6, 7, 8].

Second, deploying software of an in-house developed simulated Multi-Robot System, on the real hardware. The implementation is based on robotics middleware (ROS). ROS is an operating system, where one process has one node, and all the nodes communicate accordingly to the publish/subscribe model. Therefore, this leads to high software reuse and maintain code portability. Thus, the implementation of our robotic system is not bound to a particular simulator nor a particular real-life robot. Consequently, mitigating the system from simulation to real hardware should come with little to no implications for the rest of the system, as long as the topics that interface with the simulation-independent part of the implementation do not change.

Finally, we want to conclude with testing the system and some simple analysis of the collected data.

Working Plan

1. Familiarize yourself with programming on ROS
2. Familiarize yourself with the in-house built simulated ROS-based multi-robot system
3. Hardware set up
4. Deploy the existing software on the robots and implement the needed changes
   (a) give insights about the challenges and the issues that you have faced along the way
   (b) if needed, implemented necessary changes to make the system running
5. Test the system
   (a) collect the data and evaluate the system
   (b) compare the results with the results collected with the simulation
   (c) pursue your own ideas

Pre-requisites

- Good Python and C/C++ skills
- Previous knowledge or experience with ROS is a plus
- Interest in playing with real hardware

Deliverables

- Source code of the implementation directly usable on the real hardware
- Technical report with comprehensive documentation of the implementation, i.e. API description and usage instructions.
- Final thesis report written in conformance with TUM guidelines.

References

[1] Danny Weyns and Tanvir Ahmad. Claims and evidence for architecture-based self-adaptation: a systematic literature review. In *European Conference on Software Architecture*, pages 249–265. Springer, 2013.

[2] Stepan Shevtsov, Mihaly Berekmeri, Danny Weyns, and Martina Maggio. Control-theoretical software adaptation: A systematic literature review. *IEEE Transactions on Software Engineering*, 44(8):784–810, 2018.

[3] Henry Muccini, Mohammad Sharaf, and Danny Weyns. Self-adaptation for cyber-physical systems: a systematic literature review. In *Proceedings of the 11th international symposium on software engineering for adaptive and self-managing systems*, pages 75–81. ACM, 2016.

[4] Danny Weyns. Software engineering of self-adaptive systems: an organised tour and future challenges. *Chapter in Handbook of Software Engineering*, 2017.

[5] What is TurtleBot? `https://www.turtlebot.com/`.

[6] TurtleBot overview page (ROS). `http://wiki.ros.org/Robots/TurtleBot`.

[7] TurtleBot3 pacakge:. `https://github.com/ROBOTIS-GIT/turtlebot3`.

[8] TurtleBot 3 e-manual (specificafions):. `http://emanual.robotis.com/docs/en/platform/turtlebot3/specifications/#specifications`.

Fakultät für Informatik
Lehrstuhl 4
Software & Systems Engineering
Prof. Dr. Alexander Pretschner

Boltzmannstraße 3
85748 Garching bei München

Tel: `+49 (89) 289 - 17830`
`https://www4.in.tum.de`