The Carologistics RoboCup Logistics Team 2020

Till Hofmann[1], Sebastian Eltester[2], Tarik Viehmann[1], Nicolas Limpert[2], Victor Mataré[2], Alexander Ferrein[2], and Gerhard Lakemeyer[1]

[1] Knowledge-Based Systems Group, RWTH Aachen University, Germany
[2] MASCOR Institute, FH Aachen University of Applied Sciences, Germany

Abstract. The Carologistics team participates in the RoboCup Logistics League for the ninth year. The RCLL requires precise vision, manipulation and path planning, as well as complex high-level decision making and multi-robot coordination. We provide an overview of our approach with focus on navigation, perception, and high-level reasoning.

The team members in 2020 are David Bosen, Sebastian Eltester, Mostafa Gomaa, Till Hofmann, Nicolas Limpert, Victor Mataré, Cindy Mund, Daniel Swoboda, and Tarik Viehmann.

This paper is based on last year's team description [6] and champions paper [5].

1 Introduction

The Carologistics RoboCup Team is a cooperation of the Knowledge-Based Systems Group (RWTH Aachen University) and the MASCOR Institute (FH Aachen University of Applied Sciences). The team was initiated in 2012. Doctoral, master, and bachelor students of both partners participate in the project and bring in their specific strengths tackling the various aspects of the RoboCup Logistics League (RCLL): designing hardware modifications, developing functional software components, system integration, and high-level control of a group of mobile robots.

Our team has participated in RoboCup 2012–2019 and the RoboCup German Open (GO) 2013–2019. We were able to win the GO 2014-2019 as well as the RoboCup 2014-2017 [21,22,20,7] and 2019 [5], demonstrating flexible task coordination, robust collision avoidance and self-localization through an easily maintainable and extensible framework architecture.

In the following, we provide an overview of our system, starting with our robot platform in Section 2. In Section 3, we continue by describing our approach to path planning, before we explain our approach to perception and in particular conveyor belt detection in Section 4. In Section 5, we summarize our approach to high-level decision making, before we conclude in Section 6.

1.1 The RoboCup Logistics League

The RoboCup Logistics League (RCLL) [12] is a RoboCup [9] competition with a focus on smart factories and production logistics. In the RCLL, a team of mobile robots has to fulfill dynamically generated orders by assembling workpieces. To assemble such products, the robots operate and transport workpieces between static production machines. The major challenges of the RCLL include typical robotics tasks such as localization, navigation, perception, and manipulation, with a particular focus on reasoning tasks such as planning, plan execution, and execution monitoring.

The game is controlled by a semi-automatic referee box (*refbox*) [23]. The refbox generates dynamic orders that consist of the desired product configuration and a requested delivery time window, which must be manufactured by the robots of each team. Each requested product consists of a base piece (colored red, black, or silver), up to three rings (colored blue, green, orange, or yellow), and a cap (colored black or gray), resulting in 246 possible product configurations. The complexity of a product is determined by the number of required rings, where a C0 product with zero rings is a product of the lowest complexity, and a C3 product with three rings is a product of the highest complexity. Each team has an exclusive set of seven machines of five different types of Modular Production System (MPS) stations. To manufacture a requested product, the team has to execute a sequence of production steps by means of operating the MPS stations.

2 The Carologistics Platform

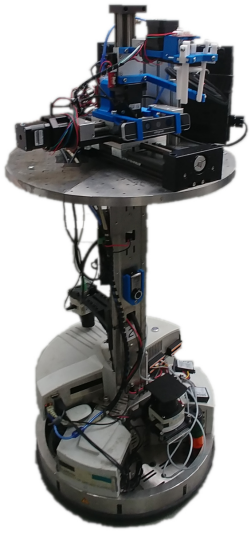


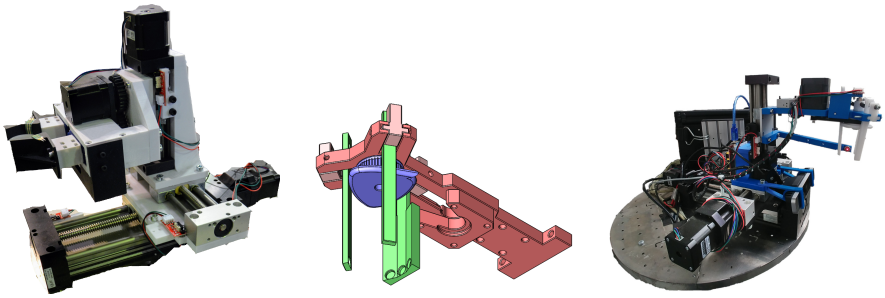
Fig. 1: The Carologistics Robotino

The standard robot platform of this league is the Robotino by Festo Didactic [8]. The Robotino is developed for research and education and features omni-directional locomotion, a gyroscope and webcam, infrared distance sensors, and bumpers. The teams may equip the robot with additional sensors and computation devices as well as a gripper device for product handling. The Carologistics Robotino is shown in Figure 1.

Sensors We use one forward-facing and one tilted, backward-facing SICK TiM571 laser scanner for collision avoidance and self-localization. Using a second laser scanner in the back allows us to fully utilize the omni-directional locomotion of the Robotino. In addition to the laser scanners, we use a webcam for detecting the MPS identification tags, and a Creative BlasterX Senz3D camera for conveyor belt detection.

2.1 Gripper System

Our gripper system consists of three linear axes and a three-fingered gripper, as shown in Figure 2. The three axes are driven by stepper motors, which allows movements with sub-millimeter accuracy. The axes are controlled by an Arduino, which in turn receives commands from the Robotino main computer.



(a) The three linear axes driven by stepper motors

(b) The CAD model of the three-fingered gripper

(c) The complete gripper system

Fig. 2: The gripper system consisting of three linear axes and a self-centering gripper with three fingers

The gripper uses three fingers and grips the workpiece from above. This allows increased robustness and precision, as the workpiece is always centered between the three spring-loaded fingers, independent of positioning errors.

2.2 Architecture and Middleware

The software system of the Carologistics robots combines two different middlewares, Fawkes [13] and ROS [24]. This allows us to use software components from both systems. The overall system, however, is integrated using Fawkes. Adapter plugins connect the systems, for example to use ROS' 3D visualization capabilities. The overall software structure is inspired by the three-layer architecture paradigm [4], as shown in Figure 3. It consists of a deliberative layer for high-level reasoning, a reactive execution layer for breaking down high-level commands and monitoring their execution, and a feedback control layer for hardware access and functional components. The communication between single components – implemented as *plugins* – is realized by a hybrid blackboard and messaging approach [13].

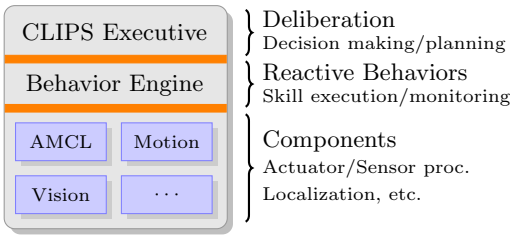


Fig. 3: Behavior Layer Separation [17]

3 Navigation

One of the most time-consuming tasks in the RCLL is navigation. From a single robot's perspective, the RCLL is a difficult multi-agent path planning scenario. The whole navigation system has to provide fast and collision free navigation with up to five other robots, only two of which can be communicated with. Hence the system has to deal with dynamic obstacles since the opponent robot's strategies are unknown. In addition, communication is unreliable, resulting in a decoupled system considering all robots on the field equally. Performing path planning in a classic single-agent approach turns out to be the best choice here.

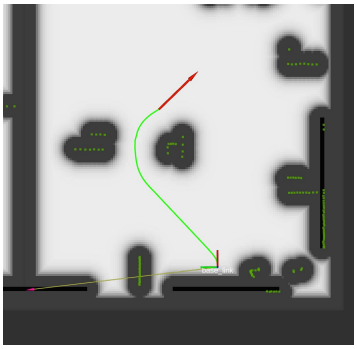


Fig. 4: Path planning in the RCLL [5]. Obstacles are represented in occupancy grids which in turn are used for the path planning environment.

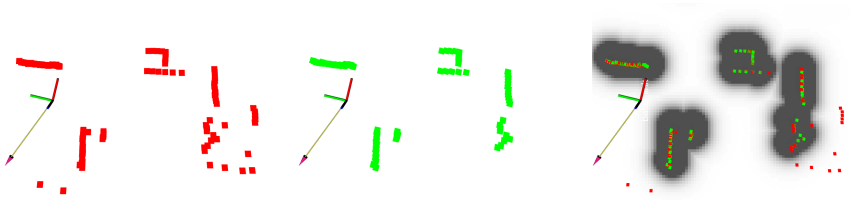
Switching to ROS Navigation [11] from a custom navigation system in 2017 has proven to be a flexible yet robust approach for single-agent pathplanning. Having been in active development for more than ten years, ROS Navigation is widely adopted on many robot platforms and is hence well-tested. As per its design, ROS Navigation is structured to separate global and local path planning. Global planning calculates shortest paths in a gradient-descent method, which does not take kinematic constraints into account. Given a feasible global path the local planner (being the controller) calculates collision free velocity commands to be executed by the mobile base. Figure 4 shows a sample screenshot with obstacles and static map components represented in an occupancy grid (costmap) [10]. Occupied cells are inflated to keep the robot from navigating too close to obstacles or walls. The shown path navigating to a position behind an MPS station reveals an important aspect of our configuration: Within narrow passages the global planner is configured to prefer paths navigating in the middle between the obstacles to allow fast motion.

The default local planner of ROS Navigation is supposed to be used for differentially driven robots. Hence it cannot leverage the Robotino's kinematic freedom that allows strafing motions. To address this we use a timed elastic band approach [26] as a controller which generates trajectories allowing the robot to save time during motion by reaching the final goal orientation even though it is still moving.

Moving obstacles are more of a threat than static obstacles due to the uncertain goals they seek. To address this issue we decided to take environmental aspects into account and filter the laser data to only consider dynamic obstacles. Since the boundaries of the playing field are part of the static map and the poses of the MPS are known during the production phase we can filter out laser beams representing static information. This helps us control the maximum velocity of

the local planner by limiting the maximum velocity to a safe value in situations where the robot is close to at least one dynamic obstacle.

A common problem with laser scanners is sensor noise. As per design the costmap can not distinguish whether a laser beam can be neglected so it has to consider each beam equally. As a result, sensor noise is also represented as full scale obstacles within the costmap, threatening the efficiency of the whole navigation system. To overcome this issue we decided to further filter the laser by only considering a laser beam to represent an obstacle if it has a set of neighboring laser beams. Figure 5 shows an example of this scenario. This increased the efficiency significantly, resulting in a robust and fast navigation solution.



(a) Raw laser data (b) Filtered laser data (c) Resulting costmap

Fig. 5: Laser filtering process [5]. 5a shows the unfiltered laser readings and the position of the robot in the bottom left. 5b depicts the filtered laser. 5c is the resulting costmap representing filtered obstacles in an overlay to the filtered and unfiltered laser beams. Note the single laser beams in the bottom left and right corners that are not taken into account.

4 Perception: Conveyor Belt Detection

Every production step in the RCLL ultimately comes down to either placing or picking a workpiece on or from a narrow conveyor belt that is only a few millimeters wider than the workpiece itself. Producing a medium-high complexity product can already involve 18 pick or place operations. Since a single manipulation error is likely to result in total loss of the product, reliability (and therefore precision) is of paramount importance.

To achieve this, we employ a three-stage pipeline, with the individual stages increasing in precision, but also in computational cost:

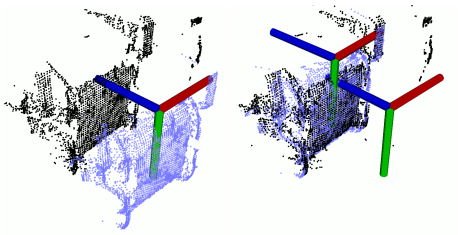


Fig. 6: The third (ICP) stage of our conveyor detection system. Left: Model pointcloud (blue) roughly aligned to scene (black) based on initial guess. Right: After running ICP, the model is aligned to the scene precisely.

Stage 1 – *RANSAC line fitting.* This stage works on the planar pointcloud generated from the SICK TiM571 laser scanners that is also used for self-localization. The RANSAC algorithm [3] searches the entire surroundings for line parametrisations that match the width of the MPS side panels. This gives us an estimate of where to look for the conveyor belt in the 3D camera picture. This stage is imprecise ($\epsilon \lesssim 10$cm), but fast (30Hz), robust and requires no initial estimate.

Stage 2 – *RANSAC plane fitting.* The estimate from the previous step gives us a rough 3D bounding box in the 3D camera picture. Here, a RANSAC searches for a rectangular plane parametrization that roughly matches the dimensions of the flat front face of the conveyor belt. This stage is as fast as the previous one, and reduces the error margin ϵ below 5mm.

Stage 3 – *ICP model fitting.* This last stage uses the Iterative Closest Point algorithm [2] to fit a model pointcloud into the scene pointcloud (cf. Figure 6). Depending on the quality of the initial estimate, this can take up to 2 seconds, but it brings the error margin ϵ down to $\lesssim 2$mm.

4.1 Workpiece Detection

Experience from the previous RoboCup has shown that using an infrared sensor to confirm a successful grip is very unreliable due to the disturbance from the 3D camera's IR pattern projector. Instead of using a special sensor we will use the images provided by the camera to confirm a successful grip. We have chosen YOLO [25] to handle the workpiece detection in the camera image. The choice is based on YOLO's real-time capabilities, which are vital to ensure a quick gripping process. The new approach will also be evaluated regarding the detection of workpieces on a Cap Station shelf. In this context YOLO's ability to reliably detect several objects in a single frame confirms our decision.

5 Behavior Engine and High-Level Reasoning

In the following we describe the reactive and deliberative layers of the behavior components. In the reactive layer, the Lua-based behavior engine provides a set of skills. Those skills implement simple actions for the deliberative layer, which is realized by an agent based on the CLIPS Executive (CX) [16], a goal reasoning framework that supports multi-agent coordination.

5.1 Lua-based Behavior Engine

In previous work we have developed the Lua-based Behavior Engine (BE) [14]. It serves as the reactive layer to interface between the low- and high-level systems. The BE is based on hybrid state machines (HSM). They can be depicted as a directed graph with nodes representing states for action execution, and/or monitoring of actuation, perception, and internal state. Edges denote jump conditions

implemented as Boolean functions. For the active state of a state machine, all outgoing conditions are evaluated, typically at about 15 Hz. If a condition fires, the target node of the edge becomes the active state. A table of variables holds information like the world model, for example storing numeric values for object positions. It remedies typical problems of state machines like fast growing number of states or variable data passing from one state to another. Skills are implemented using the light-weight, extensible scripting language Lua.

5.2 Reasoning and Planning with the CLIPS Executive

We implemented an agent based on the CLIPS Executive (CX) [16], which uses a goal reasoning model [1]. A goal describes objectives that the agent should pursue and can either *achieve* or *maintain* a condition or state. The program flow is determined by the *goal mode*, which describes the current progress of the goal. The mode transitions are determined by the goal lifecycle, which is depicted in Figure 7. When a goal is created, it is first *formulated*, merely meaning that it may be relevant to consider. The goal reasoner may decide to *select* a goal, which is then *expanded* into one or multiple plans, either by using manually specified plans or automatic planners such as PDDL planners [15]. The reasoner then *commits* to one of those plans, which is *dispatched*, typically by executing a skill of the behavior engine. Eventually, the goal is *finished* and the outcome is *evaluated* to determine the success of the goal.

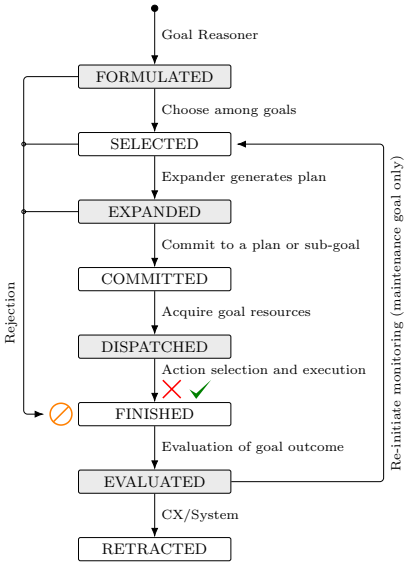


Fig. 7: The goal lifecycle with all possible goal modes [16].

The CX provides an explicit representation of the agent's world model, and its goals, plans, and actions. It separates the *domain model* with the available operators, predicates, and known facts from the *execution model*, which enhances the domain model by features that are only relevant for the execution of the plan, e.g., *exogenous actions* and *sensed predicates*. In contrast to the approaches described in [15,18], we currently do not use a planner, but instead use predefined plans.

5.3 Multi-Robot Coordination

The CX also provides means for multi-robot coordination, in particular *world model synchronization*, *mutual exclusion*, and *resource allocation* [16]. To cooperate effectively, each agent must share (parts of) its world model with the

other agents. The CX implements world model synchronization using a shared database [19,27]. Each robot uses a database instance for local (agent-specific) and global (shared) world model facts. The global world model database is synchronized as part of a replica set with the global instances of the other robots.

Based on the replicated database, the CX also implements a locking mechanism. To lock a mutex, an agent must request a *majority acknowledgement*, thereby avoiding two agents to hold the same mutex. To allow *mutual exclusion*, the CX specifies two actions *lock* and *unlock*, which may be used by the agent just as any other action. Additionally, each goal may be associated with one or multiple resources that are required in order to dispatch the goal. If one resource is currently unavailable, the goal is *rejected*. Once a goal is *retracted*, its acquired resources are released automatically.

6 Conclusion

In 2020, we are continuing the development of an agent based on the CLIPS Executive, which provides an explicit goal representation including plans and actions with their preconditions and effects. We aim to utilize the previous work on coordination mechanisms by applying explicit reasoning based on expected effects of other agent's goals in order to enhance cooperative play between the group of robots. Work on the mechanical side of the manipulator system is focused on refining the radically redesigned gripper from 2019. A core task to tackle is the optimization of the design to reduce the complexity of the assembly while maintaining a robust construction that yields a reliable performance. Our perception setup is extended by a machine learning approach to detect workpieces in our gripper and on the Cap Station shelf.

The website of the Carologistics RoboCup Team with further information and media can be found at `https://www.carologistics.org`.

Acknowledgements. We gratefully acknowledge the financial support of RWTH Aachen University and FH Aachen University of Applied Sciences.
T. Hofmann and V. Mataré were supported by the DFG grants *GL-747/23-1* and *FE-1077/4-1* (respectively) on Constraint-based Transformations of Abstract Task Plans into Executable Actions for Autonomous Robots[3].
N. Limpert was partly supported by the H2020 ROSIN project under grant agreement No 732287[4] on ROS-Industrial quality assured software components.

References

1. Aha, D.W.: Goal Reasoning: Foundations, Emerging Applications, and Prospects. AI Magazine **39**(2) (Jul 2018)
2. Chen, Y., Medioni, G.: Object modelling by registration of multiple range images. Image and vision computing **10**(3), 145–155 (1992)

[3] `http://gepris.dfg.de/gepris/projekt/288705857`
[4] `https://cordis.europa.eu/project/rcn/206395/factsheet/en`

3. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (Jun 1981)
4. Gat, E.: Three-layer architectures. In: Kortenkamp, D., Bonasso, R.P., Murphy, R. (eds.) Artificial Intelligence and Mobile Robots, pp. 195–210. MIT Press (1998)
5. Hofmann, T., Limpert, N., Mataré, V., Ferrein, A., Lakemeyer, G.: Winning the RoboCup Logistics League with Fast Navigation, Precise Manipulation, and Robust Goal Reasoning. In: RoboCup 2019: Robot World Cup XXIII. pp. 504–516. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-35699-6_41
6. Hofmann, T., Limpert, N., Mataré, V., Ferrein, A., Lakemeyer, G.: The Carologistics RoboCup Logistics Team 2019. Tech. rep., RWTH Aachen University and FH Aachen University of Applied Sciences (2019), `https://kbsg.rwth-aachen.de/~hofmann/papers/carologistics-2019-tdp.pdf`
7. Hofmann, T., Mataré, V., Neumann, T., Schönitz, S., Henke, C., Limpert, N., Niemueller, T., Ferrein, A., Jeschke, S., Lakemeyer, G.: Enhancing software and hardware reliability for a successful participation in the RoboCup Logistics League 2017. In: RoboCup Symposium – Champion Teams Track. Nagoya, Japan (2017)
8. Karras, U., Pensky, D., Rojas, O.: Mobile Robotics in Education and Research of Logistics. In: IROS 2011 – Workshop on Metrics and Methodologies for Autonomous Robot Teams in Logistics (2011)
9. Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E.: RoboCup: The Robot World Cup Initiative. In: Proc. 1st Int. Conf. on Autonomous Agents (1997)
10. Lu, D.V., Hershberger, D., Smart, W.D.: Layered costmaps for context-sensitive navigation. In: Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on. pp. 709–715. IEEE (2014)
11. Marder-Eppstein, E., Berger, E., Foote, T., Gerkey, B., Konolige, K.: The office marathon: Robust navigation in an indoor office environment. In: International Conference on Robotics and Automation (2010)
12. Niemueller, T., Ewert, D., Reuter, S., Ferrein, A., Jeschke, S., Lakemeyer, G.: RoboCup Logistics League Sponsored by Festo: A Competitive Factory Automation Benchmark. In: RoboCup Symposium 2013 (2013)
13. Niemueller, T., Ferrein, A., Beck, D., Lakemeyer, G.: Design Principles of the Component-Based Robot Software Framework Fawkes. In: Int. Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR) (2010)
14. Niemueller, T., Ferrein, A., Lakemeyer, G.: A Lua-based Behavior Engine for Controlling the Humanoid Robot Nao. In: RoboCup Symposium 2009 (2009)
15. Niemueller, T., Hofmann, T., Lakemeyer, G.: CLIPS-based execution for PDDL planners. In: ICAPS Workshop on Integrated Planning, Acting and Execution (IntEx) (2018)
16. Niemueller, T., Hofmann, T., Lakemeyer, G.: Goal reasoning in the CLIPS Executive for integrated planning and execution. In: Proceedings of the 29th International Conference on Planning and Scheduling (ICAPS) (2019)
17. Niemueller, T., Lakemeyer, G., Ferrein, A.: Incremental Task-level Reasoning in a Competitive Factory Automation Scenario. In: Proc. of AAAI Spring Symposium 2013 - Designing Intelligent Robots: Reintegrating AI (2013)
18. Niemueller, T., Lakemeyer, G., Leofante, F., Abraham, E.: Towards clips-based task execution and monitoring with SMT-based decision optimization. In: Workshop on Planning and Robotics (PlanRob) at International Conference on Automated Planning and Scheduling (ICAPS). Pittsburgh, PA, USA (Jun 2017)

19. Niemueller, T., Lakemeyer, G., Srinivasa, S.: A Generic Robot Database and its Application in Fault Analysis and Performance Evaluation. In: IEEE International Conference on Intelligent Robots and Systems (IROS) (2012)
20. Niemueller, T., Neumann, T., Henke, C., Schönitz, S., Reuter, S., Ferrein, A., Jeschke, S., Lakemeyer, G.: Improvements for a Robust Production in the RoboCup Logistics League 2016. In: RoboCup Symposium – Champion Teams Track (2016)
21. Niemueller, T., Reuter, S., Ewert, D., Ferrein, A., Jeschke, S., Lakemeyer, G.: Decisive Factors for the Success of the Carologistics RoboCup Team in the RoboCup Logistics League 2014. In: RoboCup Symposium – Champion Teams Track (2014)
22. Niemueller, T., Reuter, S., Ewert, D., Ferrein, A., Jeschke, S., Lakemeyer, G.: The Carologistics Approach to Cope with the Increased Complexity and New Challenges of the RoboCup Logistics League 2015. In: RoboCup 2015: Robot World Cup XIX (2015)
23. Niemueller, T., Zug, S., Schneider, S., Karras, U.: Knowledge-Based Instrumentation and Control for Competitive Industry-Inspired Robotic Domains. KI - Künstliche Intelligenz **30**(3), 289–299 (2016). https://doi.org/10.1007/s13218-016-0438-8
24. Quigley, M., Conley, K., Gerkey, B.P., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: an open-source Robot Operating System. In: ICRA Workshop on Open Source Software (2009)
25. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)
26. Rosmann, C., Feiten, W., Wosch, T., Hoffmann, F., Bertram, T.: Efficient trajectory optimization using a sparse model. In: European Conference on Mobile Robots (ECMR). pp. 138–143. IEEE (2013)
27. Zwilling, F.: A Document-Oriented Robot Memory for Knowledge Sharing and Hybrid Reasoning on Mobile Robots. Master's thesis, RWTH Aachen University (2017)