The Carologistics RoboCup Logistics Team 2023

Daniel Swoboda[1], Tarik Viehmann[1], Matteo Tschesche[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 eleventh 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 2023 are Tom Hagdorn, Hendrik Nessau, Matteo Tschesche, Daniel Swoboda, Tarik Viehmann, and Tim Wendt.

This paper is based on 2022's team description [22].

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 2012–2022 and the RoboCup German Open (GO) 2013–2022. We were able to win the GO 2014-2019 as well as the RoboCup 2014-2017, 2019, and 2021-2022. [15,16,14,4,3,20,22], demonstrating flexible task coordination, robust collision avoidance and self-localization through an easily maintainable and extensible framework architecture. In the 2023 season, the main focus of the team is to improve the stability and realibility of our system in terms of software and hardware.

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) [8] is a RoboCup [6] 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*) [17].

2 The Carologistics Platform

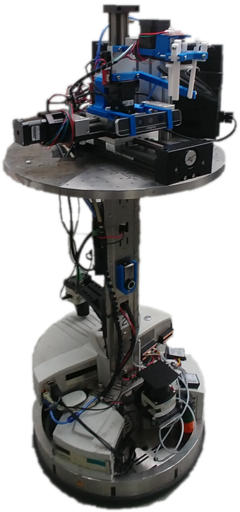


Fig. 1: The Carologistics Robotino

The standard robot platform of this league is the Robotino by Festo Didactic [5]. 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.

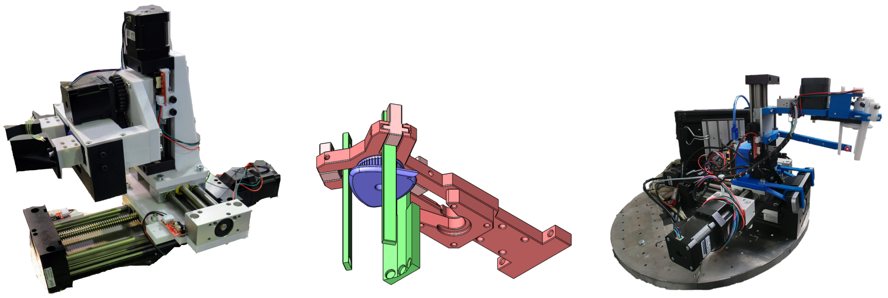
2.1 Gripper System

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. In the last RoboCup, several pre-existing bugs were found in the gripper controller. Removing them is a key issue for the team in this year's season to improve the platform stability.

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 [9] and ROS [18]. This allows us to use software components



(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

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 [2], 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 [9].

Last year we completed an experimental integration of ROS 2 into fawkes. However, while ROS 1 was providing pkg-config files for the installed software packages, which allows for easy integration in Make-based software stacks, ROS 2 rather requires modern CMake-based build systems. We took this as motivation to start upgrading fawkes to utilize CMake, which not only allows us to cleanly integrate ROS 2, but also generally greatly speeds up build times. We also packaged a significant part of ROS 2 for Fedora Linux[3], which further simplifies our setup and provides a good starting point when working with ROS 2 under Fedora.

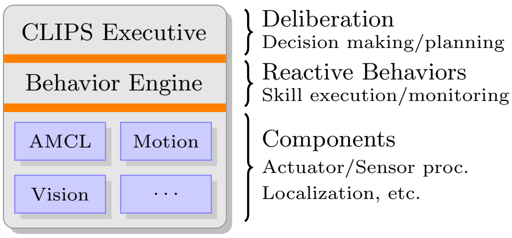


Fig. 3: Behavior Layer Separation [13]

---

[3] `https://copr.fedorainfracloud.org/coprs/tavie/ros2/`

3 Navigation

While in the past, the classic single-agent navigation system has proven to be a reliable solution for the RCLL setting, we decided to work towards a multi-agent path finding (MAPF) solution in the 2022 season [21] with the help of the ROS 2 navigation framework [7]. Our motivation is unchanged: With a MAPF approach it would be possible to handle narrow situations or intersection scenarios. In such situations, the single-agent navigation tends to oscillate because the path planner ignores dynamic obstacles. In practice, the regular RCLL field provided enough space to let the path planning of each robot find a path that overcomes such issues. Therefore the new system is supposed to combine MAPF with our existing and tested single-agent approach in case of weak network connections.

The results from the 2022 season showed that it is in principle possible to upgrade our system to the suggested MAPF approach. However, our ROS 2 integration was not sufficiently tested for deployment in the 2022 RoboCup yet, especially since we had major network issues, as also addressed in [22]. As ROS 2 is utilizing DDS, it adds another layer to our communication stack which we could not justify during the last competition. Hence we plan on upgrading our network setup with new hardware and to gather more practical experience with ROS 2 in order to finally deploy it at the official competition this year.

4 Perception

Every production step in the RCLL comes down to a pick and place task 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. While we relied on a multi-stage procedure to detect conveyor belts of the target MPS based on target pointcloud models and current depth information from our RGB-D camera, the 2022 season saw the successful integration of a RGB-based neural region of interest (ROI) detector based on the YOLO [19] framework. Our approach [22] uses a fine-tuned YOLOv4-tiny network for detection of conveyor belts, slides and workpieces. These ROI are used to calculate relative positions in a closed- loop position-based visual servoing (PBVS) task.

Integrating this new approach led to a significant increase in reliability, speed, and accuracy over the old approach used in prior seasons. However, more fine-tuning is needed to handle machine inaccuracies and minor inconsistencies in the gripper construction between the robots.

In 2023, we plan on expanding our neural-based vision to also detect machines on the shopfloor. This will serve as a pracitcal integration of the markerless-detection challenge and addition to the tag-based machine detection approach.

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) [12], 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) [10]. 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 a centralized agent based on the CLIPS Executive (CX) [12], 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 4. 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 [11]. 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.

5.3 Central Coordination

In 2022 [21] we switched from a distributed reasoning approach (i.e. one where each robot makes their own decisions based on a on a common knowledge base) to a centralized agent (i.e. one where all robots are controlled by a central instance). This decision was made with the objective in mind to move away from the incremental reasoning towards an explicit long-term production strategy. A central structure enables explicit collaboration between the robots without the overhead of the complex coordination mechanisms required to avoid conflicts in our decentral system.

The 2022 agent was developed from an agent designed to solve the challenge track of the 2021 RCLL competition. Since the main track was not part of the 2021 season, the agent needed to be extended such that it could support full games. This mainly required mechanisms for order selection and prioritization under production time constraints.

All steps required to build a product are explicitly represented through a goal tree containing the respective goals. Whenever a robot is idling, the reasoner evaluates the set of formulated goals that are currently executable on that robot and selects one among them. The orders that are pursued are selected carefully by filtering all available orders according to multiple criteria, such as estimated feasibility of attached time constraints and the required workload on each machine required to assemble the product.

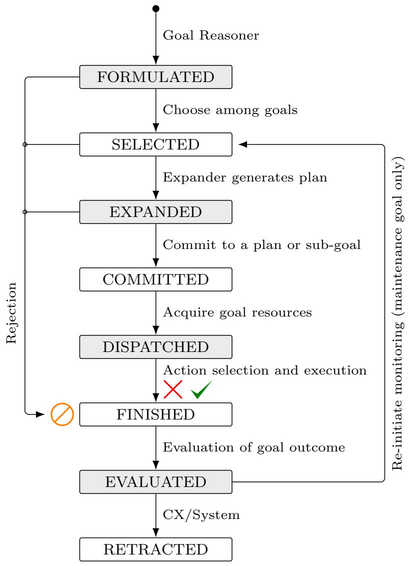


Fig. 4: The goal lifecycle with all possible goal modes [12].

While we saw that the key promises of the approach predicted last year [21] held, additional effort is required to make a more stable and reliable agent as we identified several issues in actual competition games. These issues include oversubscription (i.e., starting production of too many parallel orders), stopping production, incomplete error handling and recovery, and proper handling of robot maintenance. A centralized agent is always a single point of failure. Therefore, achieving better reliability of our agent is a key objective for us in the 2023 season.

Rule changes of the 2023 put a bigger emphasis on timely order delivery by increasing point penalties for products delivered too late. Therefore, we also plan to adapt our agent behavior, s.t. less parallel orders are being pursued and thus products are more likely to be delivered in time. To fulfill an order, two kinds of tasks need to be performed: assembly tasks and maintenance tasks. We plan on using these inherent task categories to specialize our robots. One robot will perform the sequential maintenance tasks, which allows minimal delays between the production steps. The other two robots will dynamically complete the required maintenance tasks of an order.

6 Conclusion

In 2023, we continue our work on the central agent, neural-based perception and ROS 2 integration. We focus on making our system more reliable and consistent with better error detection and handling. Additionally we focus on updating software infrastructure, increasing maintainability and future-proofing our software stack.

In order to adapt to new scoring rules, we will adapt our centralized production strategy for timely delivery by specializing robots' tasks. We still plan on deploying multi-agent path planning to ensure fast and reliable navigation in tight and narrow environments. Our existing perception pipeline will be extended with neural-based machine pose detection.

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.
This work is partially supported by the German Research Foundation (DFG) under grants *GL-747/23-1*, *2236/1* This work is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 2236/1 and Germany's Excellence Strategy – EXC-2023 Internet of Production – 390621612. 390621612.

References

1. Aha, D.W.: Goal Reasoning: Foundations, Emerging Applications, and Prospects. AI Magazine **39**(2) (Jul 2018)
2. 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)
3. 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)
4. 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 2017: Robot World Cup XXI. pp. 486–497. Springer International Publishing (2018)
5. 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)
6. 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)
7. Macenski, S., Martín, F., White, R., Clavero, J.G.: The marathon 2: A navigation system. arXiv preprint arXiv:2003.00368 (2020)
8. 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)

9. 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)
10. Niemueller, T., Ferrein, A., Lakemeyer, G.: A Lua-based Behavior Engine for Controlling the Humanoid Robot Nao. In: RoboCup Symposium 2009 (2009)
11. Niemueller, T., Hofmann, T., Lakemeyer, G.: CLIPS-based execution for PDDL planners. In: ICAPS Workshop on Integrated Planning, Acting and Execution (IntEx) (2018)
12. 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)
13. 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)
14. 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)
15. 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)
16. 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)
17. 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)
18. 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)
19. 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)
20. Viehmann, T., Eltester, S., Hofmann, T., Limpert, N., Mataré, V., Ferrein, A., Lakemeyer, G.: The Carologistics RoboCup Logistics Team 2021. Tech. rep., RWTH Aachen University and FH Aachen University of Applied Sciences (2021)
21. Viehmann, T., Limpert, N., Ferrein, A., Lakemeyer, G.: The Carologistics RoboCup Logistics Team 2022. Tech. rep., RWTH Aachen University and FH Aachen University of Applied Sciences (2022)
22. Viehmann, T., Limpert, N., Hofmann, T., Henning, M., Ferrein, A., Lakemeyer, G.: Winning the robocup logistics league with visual servoing and centralized goal reasoning. In: Eguchi, A., Lau, N., Paetzel-Prüsmann, M., Wanichanon, T. (eds.) RoboCup 2022. pp. 300–312. Springer International Publishing, Cham (2023)