The Carologistics RoboCup Logistics Team 2017

Tobias Neumann[2], Till Hofmann[1], Victor Mataré[2], Christoph Henke[3], Sebastian Schönitz[3], Tim Niemueller[1], Alexander Ferrein[2], Sabina Jeschke[3], and Gerhard Lakemeyer[1]

[1] Knowledge-Based Systems Group, RWTH Aachen University, Germany
[2] MASCOR Institute, FH Aachen University of Applied Sciences, Germany
[3] Institute Cluster IMA/ZLW & IfU, RWTH Aachen University, Germany

Abstract. The Carologistics team participates in the RoboCup Logistics League for the sixth year. As a medium complex domain balancing implementation effort and significance the league requires the development and integration of various software components. We outline the approach with an emphasis on the modifications required for recent changes to the game focusing on custom hardware and software components.

The team members in 2017 are Christoph Gollok, Mostafa Gomaa, Daniel Habering, Christoph Henke, Till Hofmann, Daniel Künster, Nicolas Limpert, Victor Mataré, Tobias Neumann, Tim Niemueller, Hannah Rost, Sebastian Schönitz, and Carsten Stoffels.

1 Introduction

The Carologistics RoboCup Team is a cooperation of the Knowledge-Based Systems Group, the IMA/ZLW & IfU Institute Cluster (both 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 all three 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 approach to the league's challenges is based on a distributed system where robots are individual autonomous agents that coordinate themselves by communicating information about the environment as well as their intended actions.

Our team has participated in RoboCup 2012–2016 and the RoboCup German Open (GO) 2013–2017. We were able to win the GO 2014-2017 as well as the RoboCup 2014 thru 2016 demonstrating flexible task coordination, robust collision avoidance and self-localization through an easily maintainable and extensible framework architecture. We have publicly released our software stack used in 2014, 2015 and 2016[4] [1].

In the following we will describe some of the challenges of the RCLL with a focus on the changes introduced in 2017. In Section 2 we give an overview

[4] Software stack available at `https://www.fawkesrobotics.org/projects/rcll2016-release/`

of the Carologistics platform and describe how we adapted the platform in the past year. We continue highlighting our behavior components in Section 4 and our continued involvement for advancing the RCLL as a whole as well as the RoboCup Industrial umbrella league in Section 5 before concluding in Section 6.

1.1 RoboCup Logistics League 2017

As in previous years, the goal is to maintain and optimize the material flow in a simplified Smart Factory scenario. Two competing groups of up to three robots each use a set of exclusive machines spread over a common playing field to produce and deliver products (cf. [3,4,5]). After the league switched from purely symbolic production to Festo's physical Modular Production System (MPS) in 2015 [4], the rules and field layout have been incrementally refined to focus on challenges that are relevant to the Industry 4.0 movement [6].

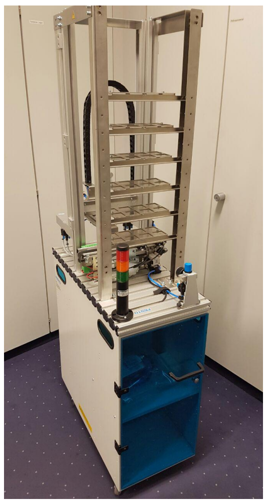


Fig. 1. The newly added Storage Station [2]

For 2017, the field layout has been redefined so that the autonomous referee box (refbox, cf., Section 5.1) is able to generate a precise machine setup. The complexity and points gain of the exploration phase have been reduced, and teams receive the true field layout at the beginning of the production phase. This way teams can completely focus on the production phase without the need to fulfill the exploration phase first to be able to gain points. A new type of MPS, the so-called Storage Station (see Figure 1 and [2]), was added as a low-risk/low-reward means to satisfy certain orders quickly.

2 The Carologistics Platform

The standard robot platform of this league is the Robotino by Festo Didactic [7]. The Robotino is developed for research and education and features omnidirectional 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.

2.1 Hardware System

Our current robot system is based on the Robotino 3. The modified Robotino used by the Carologistics RoboCup team is shown in Figure 2 and features an additional webcam to identify machine markers, a RealSense depth camera to recognize the conveyor belt, and two Sick laser range finders. We use a forward facing Sick TiM571 and a tilted backwards facing Sick TiM551 laser scanner for collision avoidance and self-localization. The TiM571 has a scanning range of 25 m (10 m for the TiM551) at a resolution of $1/3$ degrees (1 degree for the TiM551).

An additional laptop increases the computation power and allows for more elaborate methods for self-localization, computer vision, and reasoning.

Several parts were custom-made for our robot platform. As described in detail in the next section, a custom-made gripper based on Festo fin-ray fingers and 3D-printed parts are used for product handling. The gripper is able to adjust for slight lateral and height offsets. The previously used servo motors have been replaced by stepper motors in order to increase the positioning accuracy of the lateral axis. The motor is controlled with an additional Arduino board together with a motor shield. The acceleration of the motors follows an acceleration profile for smoothly increasing an decreasing the motor speed to avoid positioning errors. As no encoders are used a micro switch for initializing the lateral axis position is used.

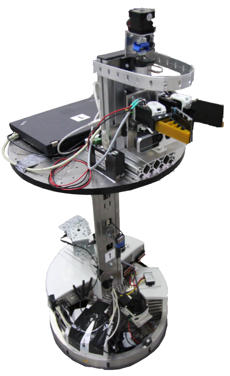


Fig. 2. Carologistics Robotino 2016 [8].

2.2 Mechanical Adaptations

To increase the agility of the robot, an additional back-facing laser scanner was added. As shown in Figure 3 on the left, the laser is mounted just above the Robotino's computational unit under a slope of 12°, enabling it to perceive other robots' main body and not only the center pillar.

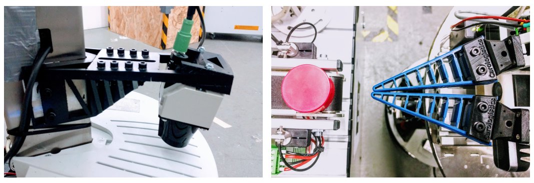


Fig. 3. Back-facing laser and new gripper

Another adaptation was made to the gripper system. Some of the grippers were damaged in past games and had to be replaced by the new version of Festo flex fingers, which required a newly designed mounting as shown in Figure 3 on the right. The design consists of two easily separable parts. The first part is attached to the gripper while the second part is attached to the Dynamixel motor. Both parts are mounted together via a dovetail guide to enable fast replacement of a damaged gripper during a match.

2.3 Architecture and Middleware

The software system of the Carologistics robots combines two different middlewares, Fawkes [9] and ROS [10]. 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 [11]. 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

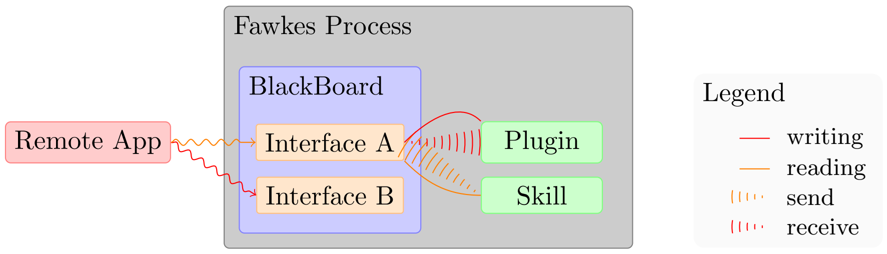


Fig. 4. Components communicate state data via interfaces stored in the blackboard. Commands and instructions are send as messages. Communication is universally shared among functional plugins and behavioral components [9].

components. The lowest layer is described in Section 3. The upper two layers are detailed in Section 4. The communication between single components – implemented as *plugins* – is realized by a hybrid blackboard and messaging approach [9]. This allows for information exchange between arbitrary components. As shown in Figure 4, information is written to or read from *interfaces*, each carrying certain information, e.g. sensor data or motor control, but also more abstract information like the position of an object. The information flow is somewhat restricted – by design – in so far as only one component can write to an interface. Reading, however, is possible for an arbitrary number of components. This approach has proven to avoid race conditions when for example different components try to instruct another component at the same time. The principle is that the interface is used by a component to provide state information. Instructions and commands are sent as messages. Then, multiple conflicting commands can be detected or they can be executed in sequence or in parallel, depending on the nature of the commands.

3 Advances to Functional Software Components

A plethora of different software components is required for a multi-robot system. In this section, we focus on changes for this year's competition, namely improvements on the MPS detection, a new path planning module and a revised conveyor belt detection.

3.1 MPS Detection and Approaching

We use a combination of tag detection and line fitting on the laser data to detect and approach an MPS. In a first step, the tag on the machine is used to validate whether the robot is approaching the correct machine and to roughly align the robot to the machine. As the tag vision only gives an imprecise position and especially rotation of the detected tag, we perform a second alignment step by searching the laser data for a suitable line, which is then used for a more precise alignment to the machine.

Markerless Machine Detection As in 2016, this year's technical challenges will include recognition of the machine type without the use of AR tags. To do so an artificial neural network (ANN) will be trained with several RGB-D pictures recorded for each machine during the Robocup GermanOpen 2017. During machine detection a picture is fed to the ANN, which then calculates a similarity measure for each possible machine type.

Since some machine types offer very few differentiating features, two thresholds are defined. The first specifies a minimum probability for the best match, while the second specifies a maximum probability for all other possible matches. To reach sufficient certainty for a successful report, both thresholds have to be satisfied for multiple perspectives on a single machine.

3.2 Path Planning

With the change in the league towards using MPS machines, the obstacles for the path planner changed tremendously. The previously used path planner, colli [12] is a reaction based, state-less path planner, which has design-based problems in coping with these new kind of obstacles and long, narrow gangways. For this reason, we decided to switch to the *Timed Elastic Band* (TEB) local planner [13], which is a planner to be used in the ROS package *move_base*. While most path planners optimize the distance of the plan, the TEB local planner optimizes the time needed to fulfill the path. Holonomic platforms are directly supported, however they can be restricted by weighing different behaviours resulting in smooth plans which can be executed fast by the Robotino platform.

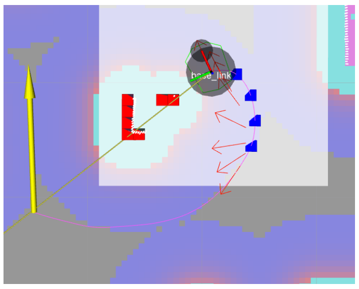


Fig. 5. Path generated by the *TEB* local planner in the ROS *move_base* package.

The topological-graph based global planner which is created based on the machines positions will also be changed this season. While we have used a Voronoi graph in the past, the new smaller zones of this years competition enabled us to use a grid based method for the generation. Doing so we can generate more parallel paths which enables us to avoid other robots better.

3.3 Conveyor Belt Detection

The conveyor belts are rather narrow compared to the products and thus require precise handling. The tolerable error margin is in the range of about 3 mm. The marker on a machine allows to determine the lateral offset from the gripper to the conveyor belt. It gives a 3D pose of the marker with respect to the camera and thus the robot. However, this requires a precise calibration of the conveyor belt with respect to the marker. While ideally this would be the same for each machine, in practice there is an offset which would need to be calibrated per station. Therefore, we are using the approach described in 3.1 for a pre-alignment, which is then improved with our depth-based conveyor detection, where a point cloud from an Intel RealSense F200 camera is used to detect the conveyor. This is done by pruning the point cloud towards our region of interest by fusing the initial guess of the belt gathered by the machine position detected with the laser scanner. Afterwards a plane search is done to detect the precise pose of the front-plane of the conveyor belt and its normal.

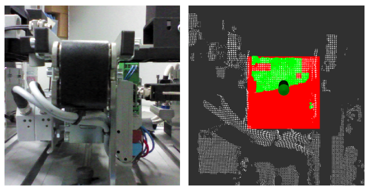


Fig. 6. Depth-based conveyor belt detection. Left: RGB picture, right: point cloud with detected conveyor belt and its normal [8].

4 High-level Decision Making and Task Coordination

The behavior generating components are separated into three layers, as depicted in Figure 7: the low-level processing for perception and actuation, a mid-level reactive layer, and a high-level reasoning layer. The layers are combined following an adapted hybrid deliberative-reactive coordination paradigm.

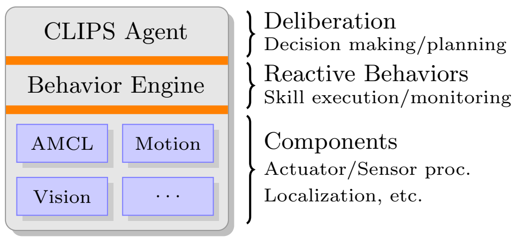


Fig. 7. Behavior Layer Separation [5]

The robot group needs to cooperate on its tasks, that is, the robots communicate information about their current intentions, acquire exclusive control over resources like machines, and share their beliefs about the current state of the environment. Currently, we employ a distributed, local-scope, and incremental reasoning approach [6]. This means that each robot determines only its own action (local scope) to perform next (incremental) and coordinates with the others through communication (distributed), as opposed to a central instance which plans globally for all robots at the same time or for multi-step plans.

In the following we describe the reactive and deliberative layers of the behavior components. For computational and energy efficiency, the behavior components need also to coordinate activation of the lower level components.

4.1 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 active state is changed to the target node of the edge. 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.

4.2 Robot Memory

In sufficiently complex robotics domains, we typically observe some degree of code duplication when common (e.g. geometric) operations are required at different levels of the framework. These operations typically revolve around an intermediate, factual knowledge level that consists of processed sensory information and agent execution state, often termed *world model*. This world model is now synchronized between robots by the generic fact representation engine *RobotMemory* [15], which is based on MongoDB. The MongoDB back-end provides an easy way to gracefully handle robot failover and reduce network traffic through incremental updates. To reduce code duplication, RobotMemory also introduces the concept of a *computable*, which is a generic operation on world model facts that can be accessed trough a MongoDB *view*. Such a view is essentially a stored (and optionally cached) DB query that can be used to combine existing MongoDB functionality with arbitrary operations offered by the robotics framework.

4.3 Reasoning and Planning

The problem at hand with its intertwined world model updating and execution naturally lends itself to a representation as a fact base with update rules for triggering behavior for certain beliefs. We have chosen the CLIPS rules engine [16], because using incremental reasoning, the robot can take the next best action at any point in time whenever the robot is idle. This avoids costly re-planning (as with approaches using classical planners) and it allows us to cope with incomplete knowledge about the world. Additionally, it is computationally inexpensive. More details about the general agent design and the CLIPS engine are in [17].

The agent for 2017 is based on the continued development effort of our CLIPS-based agent [5]. The simplified exploration phase requires a new approach in the exploration agent. Since the number of zones has increased from 24 to 104, deliberately exploring zone after zone is not feasible anymore. Consequently, the exploration strategy has been changed to a purely reactive approach, where robots roam the playing field while watching for sensor conditions that indicate a zone with an MPS in it.

We have evaluated several different possibilities for the implementation of agent programs in the RCLL [17] and are making efforts towards a centralized global planning system.

4.4 Multi-robot Simulation in Gazebo

The character of the RCLL game emphasizes research and application of methods for efficient planning, scheduling, and reasoning on the optimal work order of production processes handled by a group of robots. An aspect that distinctly separates this league from others is that the environment itself acts as an agent by posting orders and controlling the machines' reactions. This is what we call *environment agency*. Naturally, dynamic scenarios for autonomous mobile robots are complex challenges in general, and in particular if multiple competing agents are involved. In the RCLL, the large playing field and material costs are prohibitive for teams to set up a complete scenario for testing, let alone to have two teams of robots. Additionally, members of related communities like planning and reasoning might not want to deal with the full software and system complexity. Still they often welcome relevant scenarios to test and present their research. Therefore, we have created an *open simulation environment* [18,19] based on Gazebo[5].

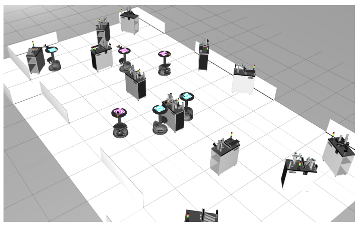


Fig. 8. Simulation of the RCLL 2015 with MPS stations [6].

5 League Advancements and Continued Involvement

We have been active members of the Technical, Organizational, and Executive Committees and proposed various ground-breaking changes for the league like merging the two playing fields or using physical processing machines in 2015 [3,4]. Additionally, we introduced and currently maintain the autonomous referee box for the competition and develop the open simulation environment described above. We have also been a driving factor in the establishment of the RoboCup Industrial umbrella league [4]. It serves to coordinate and bring closer the efforts of industrially inspired RoboCup leagues. The first steps are the unification to a common referee box system (Section 5.1) and the introduction of a cross-over challenge (Section 5.3).

5.1 RCLL Referee Box and MPS Stations

The Carologistics team has developed the autonomous referee box (refbox) for the RCLL which was deployed in 2013 [3]. It strives for full autonomy on the game controller, i.e., it tracks and monitors machine states, creates (randomized) game scenarios, handles communication with the robots, and interacts with a human referee. In 2014, the refbox has been adapted to the merged fields and two opposing teams on the field at the same time. We have also implemented a basic encryption scheme for secured communications.

[5] More information, media, the software itself, and documentation are available at `http://www.fawkesrobotics.org/projects/llsf-sim/`

In 2017, the automated machine positioning must be integrated as well as a new implementation of the new communication method between the referee box and MPS machines. Furthermore, there are efforts to create a general RoboCup Industrial Referee Box[6] based on the RCLL refbox. This would then be used in the RCLL as well as the RoboCup Industrial @Work League.

5.2 Public Release of Full Software Stack

Over the past ten years, we have developed the *Fawkes Robot Software Framework* [9] as a robust foundation to deal with the challenges of robotics applications in general, and in the context of RoboCup in particular. It has been developed and used in the Middle-Size [20] and Standard Platform [21] soccer leagues, the RoboCup@Home [22,23] service robot league, and now in the *RoboCup Logistics League* [19].

The Carologistics are the first team in the RCLL to publicly release their software stack. Teams in other leagues have made similar releases before. What makes ours unique is that it provides a complete and *ready-to-run package with the full software* (and some additions and fixes) that we used in the competition in 2014 till 2016. This in particular *includes* the complete *task-level executive* component of 2014 and 2015, that is the strategic decision making and behavior generating software. This component was typically held back or only released in small parts in previous software releases by other teams (for any league).

In addition to our software stack, we aim to release all our modifications to other software components. This includes a set of RCLL ROS packages[7], modifications of the ROS navigation stack[8], and a port of the tag tracking library Alvar to OpenCV3[9]. Furthermore, we package third-party libraries such as the Point Cloud Library (PCL) and the RealSense camera driver for the open-source operating system Fedora to allow easy installation of our software stack.

[6] Project website: `https://github.com/robocup-industrial/rci-refbox`
[7] `https://github.com/timn/ros-rcll_ros`
[8] `https://github.com/nlimpert/navigation`
[9] `https://github.com/morxa/alvar`

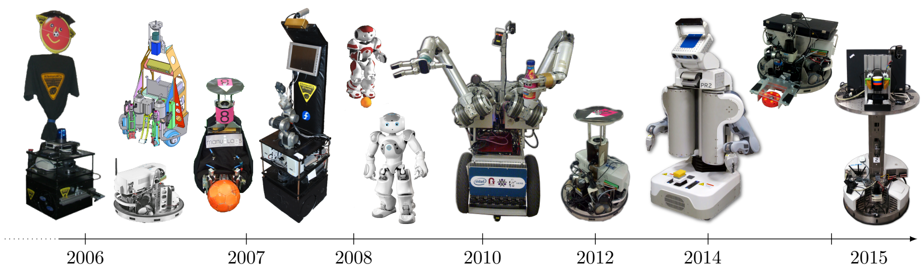
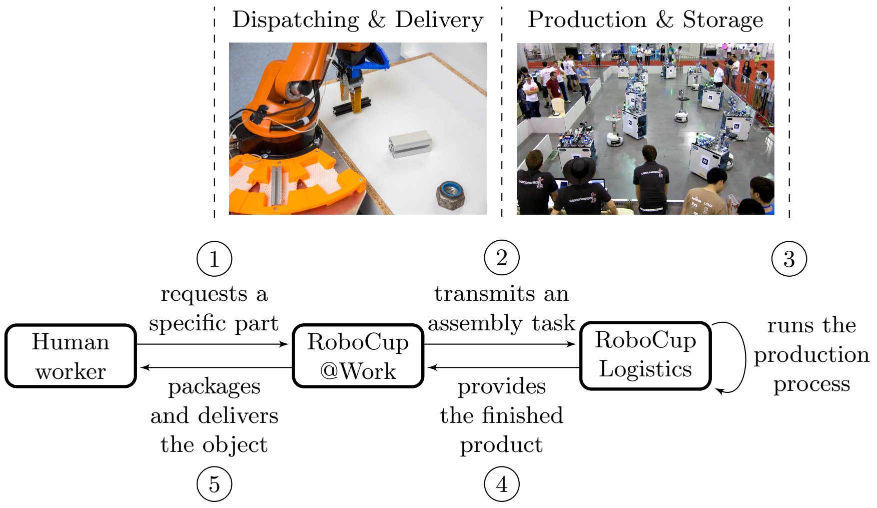


Fig. 9. Robots running (parts of) Fawkes which were or are used for the development of the framework and its components [1].

5.3 RoboCup Industrial Cross-over Challenge

As a first step for closer cooperation for the industry-inspired leagues under the RoboCup Industrial umbrella, together with stakeholders from the @Work league we have initiated a crossover challenge [24]. It describes several milestones towards closer cooperation. The task for the first year is depicted in Figure 10.



Dispatching & Delivery
Production & Storage
1
requests a specific part
2
transmits an assembly task
3
runs the production process
Human worker
RoboCup @Work
RoboCup Logistics
packages and delivers the object
provides the finished product
5
4

Fig. 10. Workflow of the cross-over scenario between @Work and the RCLL [24].

5.4 Planning Competition for Logistics Robots in Simulation

As an outcome of the presentation of the RCLL at the workshop on Planning in Robotics at the International Conference on Automated Planning and Scheduling (ICAPS) in 2015, a planning competition in simulation is being prepared. At ICAPS 2016, a tutorial was held to present the idea, gather feedback, and kickstart interested teams [25]. The competition will be held during ICAPS 2017. The particular challenges are to efficiently plan in short time with dynamic orders and temporal constraints, and to provide an effective executive for multi-robot plans. The competition will be based on the simulation developed by the Carologistics team, which is being extended to be able to run in a cluster or cloud-computing setup. The idea is to foster collaboration and exchange among the planning and robotics communities. Further information is available at `http://www.robocup-logistics.org/sim-comp`.

6 Conclusion

In 2017, we have adapted our agent to the new exploration phase and started using *RobotMemory* for world model synchronization between the agents. Furthermore, we have modified our hardware platform by using a new gripper and adding a second laser scanner, which allows us to detect obstacles behind a robot and thus permits omni-directional driving with the ROS navigation stack. We have also continued our contributions to the league as a whole through active participation in the league's committees, publishing papers about the RCLL, and implementing a crossover challenge under the RoboCup Industrial umbrella. We continued to work on the development of the simulation, which is used for the *Planning Competition for Logistics Robots in Simulation* at the ICAPS 2017. As in previous years, we have released our complete software stack including our configurations to allow new teams to join the RCLL more easily.

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

Acknowledgments. We gratefully acknowledge the financial support of RWTH Aachen University and FH Aachen University of Applied Sciences.
T. Niemueller was supported by the German National Science Foundation (DFG) research unit *FOR 1513* on Hybrid Reasoning for Intelligent Systems[10].
V. Mataré and T. Hofmann were supported by the DFG grants *FE-1077/4-1* and *GL-747/23-1* (respectively) on Constraint-based Transformations of Abstract Task Plans into Executable Actions for Autonomous Robots[11].

References

1. Niemueller, T., Reuter, S., Ferrein, A.: Fawkes for the robocup logistics league. In: RoboCup Symposium 2015 – Development Track. (2015)
2. RCLL Technical Committee: RoboCup Logistics League – Rules and Regulations 2017. Technical report, RoboCup Logistics League (2017)
3. 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)
4. Niemueller, T., Lakemeyer, G., Ferrein, A., Reuter, S., Ewert, D., Jeschke, S., Pensky, D., Karras, U.: Proposal for Advancements to the LLSF in 2014 and beyond. In: ICAR – 1st Workshop on Developments in RoboCup Leagues. (2013)
5. 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)
6. Niemueller, T., Lakemeyer, G., Ferrein, A.: The robocup logistics league as a benchmark for planning in robotics. In: 25th International Conference on Automated Planning and Scheduling (ICAPS) – Workshop on Planning in Robotics. (2015)
7. 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)
8. 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)

[10] `http://www.hybrid-reasoning.org`
[11] `http://gepris.dfg.de/gepris/projekt/288705857?language=en`

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. 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)
11. Gat, E.: Three-layer architectures. In Kortenkamp, D., Bonasso, R.P., Murphy, R., eds.: Artificial Intelligence and Mobile Robots. MIT Press (1998) 195–210
12. Jacobs, S., Ferrein, A., Schiffer, S., Beck, D., Lakemeyer, G.: Robust collision avoidance in unknown domestic environments. In Baltes, J., Lagoudakis, M.G., Naruse, T., Ghidary, S.S., eds.: RoboCup Symposium 2009. (2009)
13. Rosmann, C., Feiten, W., Wosch, T., Hoffmann, F., Bertram, T.: Efficient trajectory optimization using a sparse model. In: European Conference on Mobile Robots (ECMR), IEEE (2013) 138–143
14. Niemueller, T., Ferrein, A., Lakemeyer, G.: A Lua-based Behavior Engine for Controlling the Humanoid Robot Nao. In: RoboCup Symposium 2009. (2009)
15. Zwilling, F.: A document-oriented robot memory for knowledge sharing and hybrid reasoning on mobile robots. Master's thesis, RWTH Aachen University (2017)
16. Wygant, R.M.: CLIPS: A powerful development and delivery expert system tool. Computers & Industrial Engineering **17**(1–4) (1989)
17. Niemueller, T., Zwilling, F., Lakemeyer, G., Löbach, M., Reuter, S., Jeschke, S., Ferrein, A.: Cyber-Physical System Intelligence – Knowledge-Based Mobile Robot Autonomy in an Industrial Scenario. In: Industrial Internet of Things: Cybermanufacturing Systems. Springer (2016)
18. Zwilling, F., Niemueller, T., Lakemeyer, G.: Simulation for the RoboCup Logistics League with Real-World Environment Agency and Multi-level Abstraction. In: RoboCup Symposium. (2014)
19. 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)
20. Beck, D., Niemueller, T.: AllemaniACs 2009 Team Description. Technical report, Knowledge-based Systems Group, RWTH Aachen University (2009)
21. Ferrein, A., Steinbauer, G., McPhillips, G., Niemueller, T., Potgieter, A.: Team ZaDeAt 2009 – Team Report. Technical report, RWTH Aachen Univ., Graz Univ. of Technology, and Univ. of Cape Town (2009)
22. Schiffer, S., Lakemeyer, G.: AllemaniACs Team Description RoboCup@Home. TDP, Knowledge-based Systems Group, RWTH Aachen University (2011)
23. Ferrein, A., Niemueller, T., Schiffer, S., and, G.L.: Lessons Learnt from Developing the Embodied AI Platform Caesar for Domestic Service Robotics. In: AAAI Spring Symposium 2013 - Designing Intelligent Robots: Reintegrating AI. (2013)
24. Zug, S., Niemueller, T., Hochgeschwender, N., Seidensticker, K., Seidel, M., Friedrich, T., Neumann, T., Karras, U., Kraetzschmar, G., Ferrein, A.: An Integration Challenge to Bridge the Gap among Industry-inspired RoboCup Leagues. In: RoboCup Symposium. (2016)
25. Niemueller, T., Karpas, E., Vaquero, T., Timmons, E.: Planning Competition for Logistics Robots in Simulation. In: WS on Planning and Robotics (PlanRob) at Int. Conf. on Automated Planning and Scheduling (ICAPS), London, UK (June 2016)