Towards a Tool-based Methodology for Developing Software for Dynamic Robot Teams

Roland Glück[2], Alwin Hoffmann[1], Ludwig Nägele[1], Andreas Schierl[1], Wolfgang Reif[1] and Heinz Voggenreiter[2]

[1]*Institute for Software & Systems Engineering, University of Augsburg, 86159 Augsburg, Germany*

[2]*German Aerospace Center, Institute of Structures and Design, Center for Lightweight Production Technology (ZLP), 86159 Augsburg, Germany*

Keywords: Software Methodology, Cooperating Robots, Cyber-Physical Production System, Task Planning, Optimization.

Abstract: Considering initiatives like Industry 4.0 or the Industrial Internet of Things, robots will play an important role in intelligent factories, producing highly customized products with high variability and in small lot sizes. In this setting, complexity of planning and programming such robotic applications grows due to the drastic increase in flexibility, performance and robustness required. In this paper, we propose a tool-supported methodology for the development of control software for dynamically forming multi-functional robot teams. The main challenges for achieving this overall goal are modeling of robot team skills, techniques for automatically deriving process steps from the products' construction plans, finding allocations of those steps to possible robot teams with compatible skills and calculating collision-free execution schedules with a high degree of parallelization to improve cycle times. The proposed approach integrates process experts and automation experts on all levels. Two case studies will serve as test beds to the developed approach: production of carbon-fiber reinforced polymers and assembly of furniture.

1 INTRODUCTION

Nowadays, industrial robots play an important part in the efficient automation of production, with high flexibility, speed and precision as unique features compared to other automation principles. However, they are used mainly – for example in automotive industry – for the production of high lot sizes with relatively low variability. For more than ten years, research has been trying to make (industrial) robots efficiently usable in small-lot production (Malec et al., 2007; Pires, 2006). In this context, work has been performed both on increasing the inherent flexibility of robots, e.g., by sensor systems and associated control algorithms as described by Albu-Schäffer et al. (2007) and Finkemeyer et al. (2010), and on new methods for more efficient programming and for bringing robots into operation (Andersen et al., 2015; Neto et al., 2010). However, for the Industrial Internet of Things and Smart Factories (Kagermann et al., 2013), further steps in software development for robot systems have to be taken to provide the flexible adaptability of robots required to produce custom-tailored products with high variability in small lot sizes. In this context, it is no longer sufficient to consider an industrial robot as a single isolated entity: in fact, the requirements of an intelligent factory can only be fulfilled by dynamic and task-specific cooperation of several robots. Building robot teams with appropriate, possibly exchangeable tools leads to *multi-functional robot cells* with drastically increased flexibility, performance and robustness (Angerer et al., 2015). However, this flexibility comes with high demands on the control software of these systems: every robot has to be capable of several process steps, and must coordinate itself and precisely interact with other robots and machines. At the same time, the effort for a change of the control software has to decrease considerably to economically allow high variability and small lot-sizes of the products. These goals require new approaches and methods for software development. By building a detailed virtual model (called digital twin) of the production line and its work pieces, resources and processes, new approaches to handle the increased complexity arise. Different production teams can be formed dynamically by suitably linking dif-

Glück, R., Hoffmann, A., Nägele, L., Schierl, A., Reif, W. and Voggenreiter, H.
Towards a Tool-based Methodology for Developing Software for Dynamic Robot Teams.
DOI: 10.5220/0006884806050612
In *Proceedings of the 15th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2018) - Volume 2*, pages 605-612
ISBN: 978-989-758-321-6

ferent robots and end-effectors, depending on tasks and requirements. Each dynamic team of robots and end-effectors offers new skills to the production plant, yielding a highly flexible but also highly complex cyber-physical production system (Kagermann et al., 2013). Composing and synchronizing suitable robot teams for tasks is a scheduling and optimization problem, that should be automated as far as possible to be applicable to small batch production.

The main goal and contribution of this paper is to raise awareness for the challenges of using multi-functional robot cells for flexible production, and to propose a new methodology of software development for these cells. Before a multi-functional robot cell can be used efficiently, three main challenges have to be solved:

Challenge 1: While skills of single robots and their tools are frequently investigated (Huckaby et al., 2013; Björkelund et al., 2012; Pfrommer et al., 2015; Michniewicz and Reinhart, 2014; Stenmark and Malec, 2015), there is little research about robot teams with multi-functional or changing tools and sensor systems. To fully support robot teams, their skills have to be derived from the single robot skills of the participants, and properly described to form the foundation for flexible planning, scheduling and execution of tasks with robot teams.

Challenge 2: As a next step, the state space of all possible assignments of tasks to robots and teams as well as their temporal schedules with parallelism and collision-free synchronization is tremendously large, demanding suitable automated techniques to derive task assignment and schedules from the product or its structural design. Integrating appropriate planning and optimization algorithms based on a simulation of the multi-functional robot cell will be required to offer good resource utilization and cycle times for the resulting manufacturing plan.

Challenge 3: For execution, dynamic formation of robot teams and skills poses problems for current robot controllers. Here, the existing solutions for robot teams have to be extended to support dynamic teams, while still offering scalability (from a single robot up to the size of the multi-functional robot cell) and precision (with real-time guarantees for coordinated motions). For seamless transition from simulation to reality, ways of adapting the model of the robot cell to real world positions are required.

To meet these challenges, the overall goal of our work is to investigate and to apply *a tool-based development methodology for the control software of dynamically forming multi-functional robot teams*. To provide a highly automated methodological approach that fulfills this research goal, we aim to build a modular planning and simulation tool that supports production planning as well as robot team programming.

This paper is structured as follows: In Section 2, related approaches for planning and offline programming of robot tasks are introduced. Section 3 outlines the two application domains that will be used for evaluation, followed by a description of the four phases of the proposed approach in Section 4. Finally, a conclusion and outlook is given.

2 RELATED WORK

In classic production planning and control, robots are seen as machines with fixed, process specific capabilities encoded as robot programs, and controlled through manufacturing execution systems (MES) and programmable logic controllers (PLC). Coordination between multiple robot tasks is usually performed through a PLC in the cell, while synchronized actions such as simultaneous processing of work pieces with multiple robots is performed on the level of the robot controllers through vendor-specific extensions such as KUKA RoboTeam or ABB MultiMove (Gan et al., 2013). Robots and PLCs communicate by using binary or numeric signals and usually do not exchange any processing details such as the status of work pieces.

Many research approaches try to make this system and software structure more flexible. In the PPR approach (product, process resource) by Björkelund et al. (2012) and its extensions from the SkillPro project by Pfrommer et al. (2013), products, processes, resources and skills are modeled, including complex parameters such as work piece sizes and materials. Additionally, service oriented architectures (SOA) have been suggested to structure the software of MES (Pfrommer et al., 2015), encapsulating capabilities in vendor specific skill execution engines. An extension by Feldmann et al. (2013) adds semantic aspects to the model of functions and parameters, and uses SysML to model the behavior of devices and processes. During operation, appropriate devices are selected based on the abstract model, and PLC code in a language defined in IEC 61131-3 is generated. However, these approaches do not cover the cooperation of multiple robots and the forming of dynamic robot teams.

Other approaches concentrate on real time critical execution of robot actions and the cooperation of multiple robots, such as the OROCOS project (Smits et al., 2009), aRD(x) (Hammer and Bäuml, 2014) and Finroc (Reichardt et al., 2013). The iTaSC approach (Vanthienen et al., 2013) allows modeling robot

motions through constraints on the kinematic chain, which is also applicable to robot teams. These approaches however do not explicitly model robot capabilities that can be used on the MES level to plan with capabilities and constraints of robot teams.

In the field of software architectures and programming methods for industrial robots, the Robotics API (Angerer et al., 2013) allows implementing real-time critical control software for industrial robots using modern programming languages and tools. Together with a real-time robot control core, it allows simulating and directly executing synchronized operation of multiple industrial robots by different vendors (e.g. KUKA, Staubli, UR). Additionally, it puts special emphasis on consistent design principles for heterogeneous robot systems, and at run time works with an extensive model of work pieces and the robot cell. While not addressing planning or optimization problems, the concepts of modeling and executing multi-robot applications seem highly appropriate for our planned approach.

A further approach is the project ROS-Industrial[1]. While the Robot Operating System (ROS)[2] is an open-source middleware and framework for robotics, ROS-Industrial aims to enable ROS for industrial robotic applications previously technically infeasible. It defines standardized interfaces for controlling industrial robots and developing hardware-agnostic software for industrial applications. Many robot manufactures are involved such as ABB, Adept, Comau, Fanuc, Universal Robot and Motoman. Although, the distributed nature of ROS seems appropriate for multi-robot applications, it is not the focus of ROS-Industrial so far.

During off-line programming, robot programs are created in a simulation environment, avoiding occupying the robot cell during programming. However, the simulation has to be similar to reality so that the developed software can be transferred to the real cell. According to Gan et al. (2013), typical off-line programming environments include Tecnomatix RoboCad by Siemens, DELMIA Robotics Offline Programming by Dassault Systèmes and CimStation Robotics by AC&E. They concentrate on typical robot tasks such as spot or arc welding, cutting, milling or painting, and typically do not support to program dynamic or heterogeneous robot teams. Other off-line programming environments are offered by robot manufacturers, such as KUKA.Sim by KUKA or Robot Studio by ABB. However support for robot cooperation is still limited. All these off-line programming environments transform robot paths (e.g. from CAD data) into vendor specific robot code, losing information about the work pieces so that errors found during testing in the real robot cell cannot be transferred back to the model. These environments also are limited to the capabilities of the underlying robot controllers. In the academic context, simulations such as V-Rep (Rohmer et al., 2013) or Gazebo (Koenig and Howard, 2004) are typically used. However, these only support simulation, but no trajectory planning or robot programming, and thus do not offer an approach to integrate task and motion planning for robot teams.

Looking at the field of process planning and scheduling, some approaches concentrate on discrete planning algorithms for task allocation to robots. Here, planning domain definition language (PDDL) introduced by McDermott et al. (1998) and based on the STRIPS approach (Fikes and Nilsson, 1971) is often used, e.g. in the SkillPro project. Further extensions such as the one introduced by Benton et al. (2012) support soft and hard deadlines and non-linear cost functions, as well as qualitative aspects such as special treatment of fragile parts. Other approaches include the use of SAT solvers (Imeson and Smith, 2014). However, all these works are limited to a single robot that will execute the planning results, and do not support cooperative execution by a team of robots.

In recent years, the integration of task and motion planning has received increased attentions. In the work of Kaelbling and Lozano-Pérez (2011) and Levihn et al. (2013), planning and task decomposition is only performed to a limited depth, then executing the results and continuing planning based on the resulting situation to make the planning complexity manageable. In contrast, Bhatia et al. (2011) propose to integrate the two planning layers (discrete for tasks and sampling-based for motions), and plan both layers before execution. To describe assembly tasks, assembly graphs (Stenmark and Malec, 2015) offer an approach and a framework to plan, schedule and execute tasks using an industrial tool chain. Further extensions such as contact state graphs (Xiao and Ji, 2001) allow force controlled manipulation. Macho et al. (2016) introduced a hierarchical approach for modular planning and automatic programming of robot tasks in the field of carbon fiber production, incorporating expert knowledge through process specific modules and asking the user about geometric degrees of freedom that are not constrained by the problem definition. Still, these approaches do not address multi-robot planning and cooperative execution.

The field of manufacturing with multiple robots has also been covered by research approaches: Tercio (Gombolay et al., 2013) is an algorithm to assign tasks to multiple similar robot systems that work

[1] www.rosindustrial.org

[2] www.ros.org

on the same part, avoiding collisions by reducing the overlap of workspaces. Another approach by Yun and Rus (2014) uses two different robot types to plan the assembly of components from many parts by first choosing an optimal placement of assembly robots, and then planning which assembly robot each delivery robot should transport the different parts to. Knepper et al. (2013) describe the manufacturing of furniture using a team of mobile robots using an object oriented extension of PDDL, but limited to a fixed set of possible robot cooperations required for the task and not covering geometric motion planning. In contrast, an approach for the manufacturing of large structures by Bourne et al. (2015) connects planning and scheduling of robot teams, using a dynamic scheduler based on assembly graphs generated during planning, and focusing on reaching the required precision through a team of mobile robots. However, these approaches do not cover the capabilities of dynamic robot teams.

To summarize, the full potential of multi functional robot cells in intelligent factories can only be used by forming dynamic, task specific robot teams. However, this field has not been covered, neither from a production nor software point of view. Existing approaches are not up to the resulting complexity of modeling robot teams, planning and scheduling tasks and simulating and executing the results with teams of robots, and especially to the integration of all levels required for a consistent methodological approach.

3 CASE STUDIES

To evaluate and demonstrate the methodology, two case studies – one in the domain of carbon fiber-reinforced polymer production and another in the domain of furniture assembly – will serve as test beds.

3.1 Production of CFRP Parts

In aerospace industry, carbon fiber-reinforced polymers (CFRP) are a widely used material due to their strength-to-weight ratio. One possibility to produce CFRP parts is to drape multiple layers of carbon fiber cut pieces into a mold and infuse the finished composition with a suitable resin. After hardening in an oven or autoclave, CFRP parts in the desired form are obtained. Figure 1 shows a cross section through a mold and two layers of cut pieces.

Experts use specialized CAD programs to construct CFRP parts considering static and structural goal properties. This design process yields a so-called

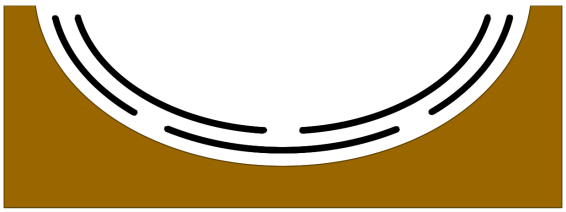

Figure 1: Example layer construction in a mold: five cut pieces in two stacked layers.

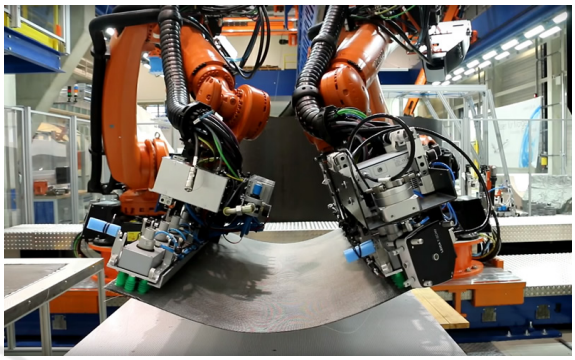

Figure 2: Two robots carrying a cut piece.

construction part design (CPD) which is used as construction plan. For automated production, one or two robots use specific end-effectors to pick cut pieces provided on a desk and place them into a form. Figure 2 shows two robots executing this task.

However, manually creating a program for robot teams based on a CPD is very tedious and can take up to hours for one single cut piece (Nägele et al., 2015). For the production of a full-scale part, hundreds of cut pieces have to be processed. To make this process economically feasible, automated or computer aided generation of programs is required.

3.2 Assembly

Geisberger and Broy (2015) describe how a cyber-physical system can handle orders of custom-tailored kitchens. Because the assembly of specific articles of furniture is a suitable task for robot teams, we also plan to apply our methodology to this domain. Here, the number of basic parts is less than in the previous case, but the process steps are of a much higher variability. Depending on the parts, assembling, screwing or gluing steps have to be executed, each of them demanding own tools and algorithms for execution and planning. Greater and heavier parts like table plates have to be handled by more than one robot. In a multi-functional cell, many steps can be performed in parallel, e.g. the simultaneous bolting of table legs or the insertion of shelves as depicted in Figure 3. In contrast to CFRP production, typically no machine-

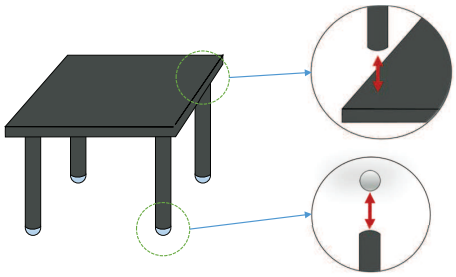

Figure 3: Example for assembly steps of a table.

readable construction plan exists for these assembly tasks. However, existing assembly instructions offer a good basis for it.

4 APPROACH

To make the complexity of dynamic robot teams manageable, we propose a planning approach with four phases as shown in Figure 4. First, an existing blueprint is analyzed and decomposed into atomic subtasks (goals) that can be solved using the existing robot team capabilities of the multi-functional cell (Section 4.1). Appropriate planning modules, which use robot team capabilities to create processes, are assigned to these goals (Section 4.2). In a scheduling and planning phase (Section 4.3), these planning modules are used to derive a consistent overall process that can be simulated and executed using a real robot cell (Section 4.4). During important steps of the entire process, expert knowledge is integrated, including process experts contributing to the top-down view of the product centered manufacturing problem, and automation experts providing a bottom-up view based on robot team capabilities. To find appropriate process parameters, the approach allows iterative parameter tuning while keeping previous results, and evaluate the effects in simulation.

4.1 Analysis

The methodology is based on a detailed blueprint describing the layout and structure of the desired product, including details about different production steps as well as temporal dependencies in the production process. An extensible set of analysis modules transforms the blueprint into a set of intermediate goals, which are analyzed further until atomic goals are reached. In the case of CFRP production, a ply-book as part of the CPD is first separated into the different layers, and then into separate cut pieces that have to be placed. Finally, for each cut piece atomic goals to pick up or drape are created. For these resulting goals, usually temporal dependencies have to be met: In the CFRP example, a cut piece may only be draped after the underlying layer has been placed. These dependencies are expressed through constraints concerning the goals. As analysis modules depend on the structure of the blueprint, they are mostly domain specific. Thus, this phase heavily depends on the domain knowledge of experts, and the analysis modules can vary in number and structure. Additionally, different analysis modules can exist for the same problem (e.g. to place the cut pieces from the center to the border, or from one side to the other), leading to alternatives that can be considered in the following phases.

As a result, the analysis provides a set of hierarchically structured, partially alternative goals, as well as constraints between the different goals.

4.2 Assignment

In the second phase, an extensible set of planning modules is used to find solutions for each of the goals identified in phase 1. Each planning module knows which types of goals it can plan for, and uses the capabilities of the robot cell to compute a specific solution. In the example of CFRP production, a planning module uses the robot capabilities 'deform', 'position' and 'fixate' to plan for the specific goal 'place cut piece'. On this layer, the main problems belong to the automation domain, so automation experts have to contribute to the creation and development of planning modules based on robot team capabilities. As a result of this phase, each goal receives a set of potentially suitable planning modules that find their required capabilities in the robot cell, as well as constraints concerning the capabilties derived by the planning modules (e.g. grasping objects of a certain size).

4.3 Scheduling and Planning

The following phase attempts to create specific and integrated robot programs based on the results of the previous phases. Thus, for each specific goal a program solution has to be calculated that is executable in the robot cell. For each goal, a planning module is chosen, and for each of its required capabilities a suitable robot or team is selected. Using the selected robot or team, the module plans and creates a specific solution that occupies the assigned robots or teams. Additionally, further constraints arise during planning, e.g. through occupied areas in the workspace. These constraints have to be considered by planning modules for following goals. If the required degrees of freedom do not allow planning the following goals, parts of the previously created pro-

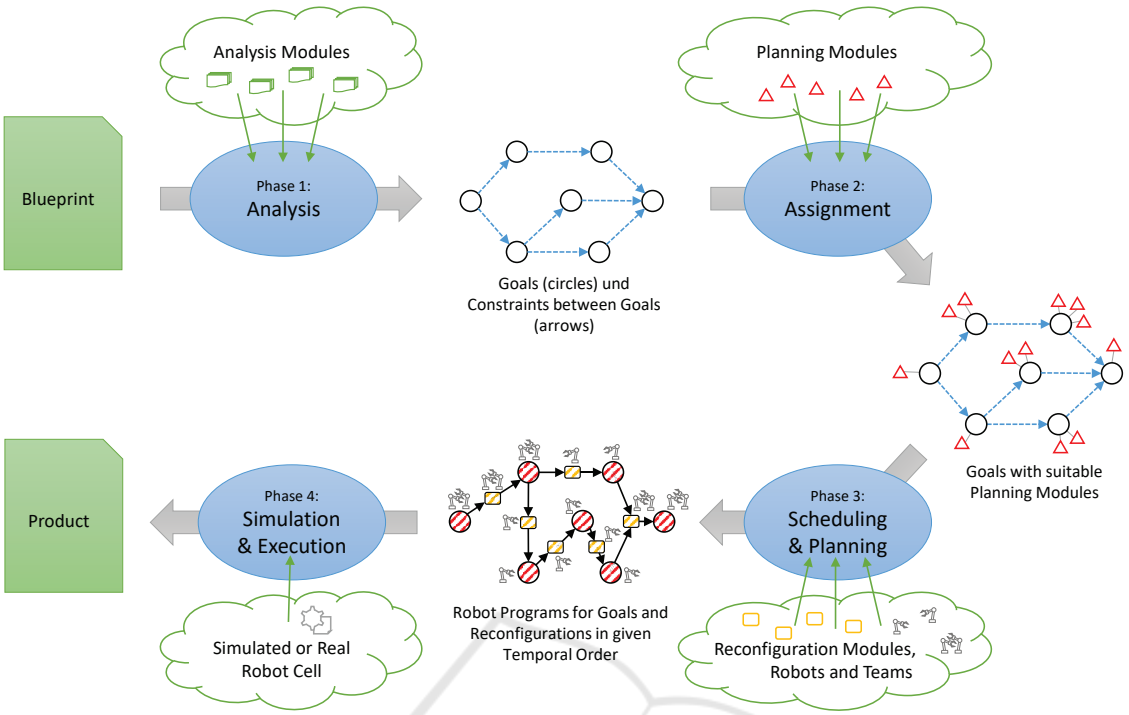

Figure 4: Four phases of the proposed approach and intermediate artifacts.

gram have to be replaced. This can be done by the same planning module by computing another solution with different constraints (e.g. using other trajectories or other robots), by choosing another planning module, or by using alternative goals found in the previous phases.

Apart from the actual task execution, each program solution for a goal contains implicit assumptions about the start and end configuration of the robot cell, in the form of an expected position of all robots before the task, as well as a resulting position after execution. To consistently match subsequent program parts, intermediate programs have to be planned that perform necessary reconfigurations in the robot cells, such as transfer motions, tool changes or dynamically creating robot teams. In the example of Figure 2, both robots can either work independently, or form a dynamic team with cooperative end-effector movements. These kinds of program completions mainly consist of actions in the robot cell and are provided by automation experts as reconfiguration modules.

The complete process can be considered as a complex and dynamic planning and optimization problem, providing a rich set of solution tools. For goals, the relevant constraints can be transformed into a formal constraint satisfaction problem that can be solved through techniques of constraint programming and optimization. The central task is to select alternative goals and specific planning modules, to assign robot teams and to calculate the temporal sequence of goals and reconfigurations. Some decisions – such as selecting alternative goals or supplying missing information – can interactively be taken by experts through semi-automatic programming, allowing automation and process experts to influence the kind and quality of results. As a result of this phase, a consistent and integrated control program has been created that reaches the given goals using robots and teams in a specific temporal order.

4.4 Simulation and Execution

Based on the created control program, process experts can use a simulation environment to test or adapt the generated program in a virtual environment, and to visualize different alternatives for comparison, while seeing meta information about process steps and formed teams. Additionally, the simulation environment allows automation experts to use advanced functions such as collision detection or geometric relationships to implement planning and reconfiguration modules, directly using the model of the robot cell for planning. To execute a control program on a real robot cell, the required robot team capabilities have to be transformed into equivalent robot and tool actions. Here, the real-time critical implementation of team cooperation, as well as challenges of distribution and decentralization have to be respected.

After a program has been tested in simulation, seamless transition to real multi-functional robot cells is required. To facilitate this, adaptations of the program need to be possible to fix observed robot misbehavior or production inaccuracies. The approach iteratively allows for adjusting and calibrating process parameters and robot cell settings such as positions or dimensions that can be transformed into a corrected control program through re-executing phase 3.

5 CONCLUSIONS

In this paper, we motivated the need for multi-functional robot cells to handle small lot sizes and mass customization. To cope with the great adaptability and flexibility of such cells and to make production economically feasible, effective planning tools are required. As a solution, we proposed a tool-supported methodology for the development of control software for dynamically forming multi-functional robot teams. The approach addresses the challenges of modeling robot team skills, automatically deriving process steps from the products' construction plans, finding allocations of those steps to possible robot teams with compatible skills and calculating appropriate execution schedules with good resource utilization and cycle times. For challenging domains, the approach allows experts from both process and automation to contribute their knowledge in all steps of process definition.

We plan to implement the proposed approach and to evaluate it in simulation as well as in reality. However, especially in the area of composing dynamic robot teams and finding optimal task schedules for them, further research is required to successfully apply the approach to challenging domains. Nevertheless, the proposed approach is a promising solution for automatic production planning in multi-functional robot cells involving expert knowledge.

ACKNOWLEDGEMENTS

This work is partly funded by the German Research Foundation (DFG) under the TeamBotS grant.

REFERENCES

Albu-Schäffer, A., Ott, C., and Hirzinger, G. (2007). A unified passivity-based control framework for position, torque and impedance control of flexible joint robots. *Intl. J. Rob. Research*, 26(1):23–39.

Andersen, R. H., Dalgaard, L., Beck, A. B., and Hallam, J. (2015). An architecture for efficient reuse in flexible production scenarios. In *Proc. 2015 IEEE Conf. on Autom. Science and Engineering (CASE)*, pages 151–157.

Angerer, A., Hoffmann, A., Schierl, A., Vistein, M., and Reif, W. (2013). Robotics API: Object-oriented software development for industrial robots. *Softw. Eng. for Robotics*, 4(1):1–22.

Angerer, A., Vistein, M., Hoffmann, A., Reif, W., Krebs, F., and Schönheits, M. (2015). Towards multi-functional robot-based automation systems. In Filipe, J., Madani, K., Gusikhin, O. Y., and Sasiadek, J. Z., editors, *Proc. 12th Intl. Conf. on Inform. in Control, Autom. & Robot. (ICINCO 2015)*, pages 438–443. SciTePress.

Benton, J., Coles, A., and Coles, A. (2012). Temporal planning with preferences and time-dependent continuous costs. In *Proc. 22nd Intl. Conf. on Autom. Planning & Scheduling, ICAPS*.

Bhatia, A., Maly, M. R., Kavraki, L. E., and Vardi, M. Y. (2011). Motion planning with complex goals. *IEEE Robotics Automation Magazine*, 18(3):55–64.

Björkelund, A., Bruyninckx, H., Malec, J., Nilsson, K., and Nugues, P. (2012). Knowledge for intelligent industrial robots. In *AAAI Spring Symposium: Designing Intelligent Robots*.

Bourne, D., Choset, H., Hu, H., Kantor, G., Niessl, C., Rubinstein, Z., Simmons, R., and Smith, S. (2015). Mobile manufacturing of large structures. In *Proc. 2015 IEEE Intl. Conf on Robotics & Automation (ICRA)*, pages 1565–1572.

Feldmann, S., Loskyll, M., Rosch, S., Schlick, J., Zuhlke, D., and Vogel-Heuser, B. (2013). Increasing agility in engineering and runtime of automated manufacturing systems. In *Proc. 2013 IEEE Intl. Conf. on Industrial Technology (ICIT)*, pages 1303–1308.

Fikes, R. and Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artif. Intell.*, 2(3/4):189–208.

Finkemeyer, B., Kröger, T., and Wahl, F. (2010). The adaptive selection matrix - a key component for sensor-based control of robotic manipulators. In *Proc. 2010 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3855–3862.

Gan, Y., Dai, X., and Li, D. (2013). Off-line programming techniques for multirobot cooperation system. In *Intl. J. of Advanced Robotic Systems*. InTech Europe.

Geisberger, E. and Broy, M. (2015). *Living in a networked world. Integrated research agenda Cyber-Physical Systems (agendaCPS) (acatech STUDY)*. Herbert Utz Verlag.

Gombolay, M., Wilcox, R., and Shah, J. (2013). Fast scheduling of multi-robot teams with temporospatial constraints. In *Proc. of Robotics: Science and Systems (RSS)*, Berlin, Germany.

Hammer, T. and Bäuml, B. (2014). The communication layer of the ardx software framework: Highly performant and realtime deterministic. *Intelligent & Robotic Systems*, 77(1):171–185.

Huckaby, J., Vassos, S., and Christensen, H. I. (2013). Planning with a task modeling framework in manufacturing robotics. In *Proc. 2013 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 5787–5794.

Imeson, F. and Smith, S. L. (2014). A language for robot path planning in discrete environments: The tsp with boolean satisfiability constraints. In *Proc. 2014 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 5772–5777.

Kaelbling, L. P. and Lozano-Pérez, T. (2011). Hierarchical task and motion planning in the now. In *Proc. 2011 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 1470–1477.

Kagermann, H., Wahlster, W., and Helbig, J., editors (2013). *Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0 – Abschlussbericht des Arbeitskreises Industrie 4.0*. acatech – Deutsche Akademie der Technikwissenschaften e.V.

Knepper, R. A., Layton, T., Romanishin, J., and Rus, D. (2013). Ikeabot: An autonomous multi-robot coordinated furniture assembly system. In *Proc. 2013 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 855–862.

Koenig, N. and Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Proc. 2004 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 2149–2154.

Levihn, M., Kaelbling, L. P., Lozano-Pérez, T., and Stilman, M. (2013). Foresight and reconsideration in hierarchical planning and execution. In *Proc. 2013 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 224–231.

Macho, M., Nägele, L., Hoffmann, A., Angerer, A., and Reif, W. (2016). A flexible architecture for automatically generating robot applications based on expert knowledge. In *Proc. 47th Intl. Symp. on Robotics, ISR 2016, Munich, Germany*. VDE Verlag.

Malec, J., Nilsson, A., Nilsson, K., and Nowaczyk, S. (2007). Knowledge-based reconfiguration of automation systems. In *Proc. 2007 IEEE Intl. Conf. on Automation Science and Engineering (CASE)*, pages 170–175.

McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., Weld, D., and Wilkins, D. (1998). PDDL — the planning domain definition language. Technical Report TR 98 003/DCS TR 1165, Yale Center for Computational Vision and Control.

Michniewicz, J. and Reinhart, G. (2014). Cyber-physical robotics – automated analysis, programming and configuration of robot cells based on cyber-physical-systems. *Procedia Technology*, 15:566–575.

Nägele, L., Macho, M., Angerer, A., Hoffmann, A., Vistein, M., Schönheits, M., and Reif, W. (2015). A backward-oriented approach for offline programming of complex manufacturing tasks. In *Proc. 6th Intl. Conf. on Automation, Robotics and Applications (ICARA 2015)*, pages 124–130.

Neto, P., Norberto Pires, J., and Paulo Moreira, A. (2010). High-level programming and control for industrial robotics: using a hand-held accelerometer-based input device for gesture and posture recognition. *Industrial Robot*, 37(2):137–147.

Pfrommer, J., Schleipen, M., and Beyerer, J. (2013). PPRS: production skills and their relation to product, process, and resource. In *Proc. IEEE 18th Conf. on Emerging Technologies & Factory Automation (ETFA 2013)*.

Pfrommer, J., Stogl, D., Aleksandrov, K., Navarro, S. E., Hein, B., and Beyerer, J. (2015). Plug & produce by modelling skills and service-oriented orchestration of reconfigurable manufacturing systems. *Automatisierungstechnik*, 63(10):790–800.

Pires, J. N. (2006). Robotics for small and medium enterprises: control and programming challenges. *Industrial Robot*, 33(6).

Reichardt, M., Föhst, T., and Berns, K. (2013). On software quality-motivated design of a real-time framework for complex robot control systems. In *Proc. 7th Intl. Workshop on Software Quality and Maintainability, Genova, Italy*.

Rohmer, E., Singh, S. P., and Freese, M. (2013). V-rep: A versatile and scalable robot simulation framework. In *Proc. 2013 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 1321–1326.

Smits, R., Laet, T. D., Claes, K., Soetens, P., Schutter, J. D., and Bruyninckx, H. (2009). Orocos: A software framework for complex sensor-driven robot tasks. *IEEE Rob. & Autom. Mag.*

Stenmark, M. and Malec, J. (2015). Knowledge-based instruction of manipulation tasks for industrial robotics. *Robotics and Computer-Integrated Manufacturing*, 33:56–67.

Vanthienen, D., Klotzbuucher, M., De Schutter, J., De Laet, T., and Bruyninckx, H. (2013). Rapid application development of constrained-based task modelling and execution using domain specific languages. In *Proc. 2013 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 1860–1866.

Xiao, J. and Ji, X. (2001). Automatic generation of high-level contact state space. *Rob. Research*, 20(7):584–606.

Yun, S. and Rus, D. (2014). Adaptive coordinating construction of truss structures using distributed equal-mass partitioning. *IEEE Trans. on Robotics*, 30(1):188–202.