Assisted Planning and Setup of Collaborative Robot Applications in Modular Production Systems

Michael Wojtynek, Julian Leichert, Sebastian Wrede
Research Institute for Cognition and Robotics (CoR-Lab)
Bielefeld University, Universitätsstr. 25,
33615 Bielefeld, Germany
{mwojtynek, jleichert, swrede}@cor-lab.uni-bielefeld.de

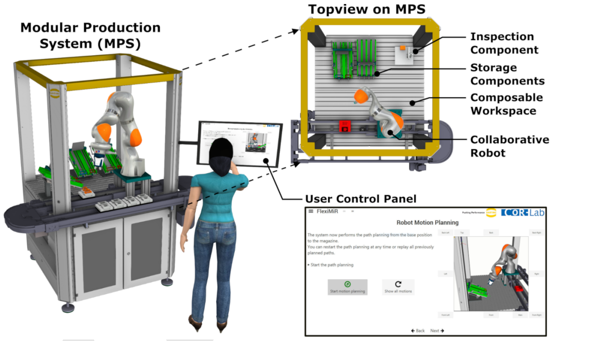
***Abstract*—Modern production requires flexible systems to fulfill customer desires of highly specialized products down to a batch size of one. Collaborative robots are considered as the key element for a versatile and flexible production in a state-of-the-art and modular production environment. Although modularization is a well known concept since the 80s, the composition of modular fabrication units and the flexible programming of robots are still challenging and depend strongly on the experience of the industrial engineer. For this reason, we present an assisted approach for interactive workspace setup targeting the operator to plan and commission efficiently applications containing collaborative robots in flexible production systems. As a central contribution, we introduce a planning workflow to take the human-in-the-loop for the workspace configuration of the robot and the arrangement of components in the modular production system. We evaluate our contribution quantitatively in an initial experimental setup performing the automatized layout in complex configurations to showcase its effectiveness.**

***Index Terms*—Flexible Robotic System, Intelligent Manufacturing, Modular Production, Workspace Reconfiguration, Industry 4.0**

I. Introduction

Small and medium-sized enterprises (SME) have been mostly robot-free so far, but collaborative robots (co-bots) are becoming increasingly popular in SME shopfloors. The most recently published numbers on industrial applications have shown a growth of 23% for collaborative robot installation from 2017 to 2018 [1]. Affordable prices, more intuitive ways of programming, and the promotional promise for a flexible utilization in modern applications make collaborative robots competitive to special purpose handling and assembly machines. But the study shows as well that the amount of collaborative robots is still small (share of just 3.24%) compared to overall installed robots (422.000 pieces). Reasons for this effects can be explained that the realization of industrial assembly tasks and the setup or re-configuration, including the robot programming, is still a non-trivial, tedious, and time-consuming process.

Installing a robot in an industrial application, where the space is precious and the task requirements are highly demanded, leads inevitably to the problem statement expressed by the following questions: *Where to place the robot in my workspace? Where to place the peripheral components containing my workpieces? Can my robot reach, grasp and manipulate these objects?* The limited operating distance of collaborative robots (approx. 1-meter radius) is one of the most impactful problems for these questions. The operating area is even more restricted near the base or outer arm distance, where the working points are hard to reach, or manipulation is impossible due to the robot kinematics and singularities. Easy record-and-play functions, provided by the manufacturers, may be sufficient for simple pick-and-place operations in the free-space. However, complex industrial applications need an efficient and computer-aided approach for workspace design and layout.



The system now performs the path planning from the base position to the magazine.
You can restart the path planning at any time or replay all previously planned paths.

▸ Start the path planning

Fig. 1: Modular Production System with a collaborative robot and exemplary components.

As the number of sold and installed (collaborative) robots in the shopfloors increases, so does the number of novice operators. The success or failure of a robot installation is strongly dependent on the intuition and knowledge of the operator. In the year 1985, Yoshikawa has already identified that the ”determination of the [robot] posture [..] for performing a given task [has] been done largely on the basis of experience and intuition” [2]. To this end, the author stated that there is need for a ”quantitative measure” to improve the robot design stages, manipulation and task planning.

The target of this contribution is to lower the entry barrier for novice robot users regarding complex assembly scenarios in limited workspaces. Skilled operators benefit from a faster robot programming and layout process. A dedicated simulation supports the operator with a guided and assisted process, takes the operator’s experience and decisions into account, and expresses the design of robot workspace by a score to advise possibilities of improvement. The operator transfers the simulated workspace layout into the real application and

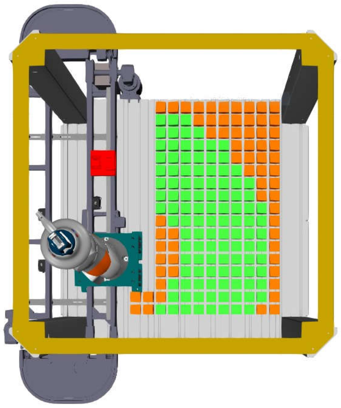


Fig. 2: Internal Reachability Analysis of the Interactive Workspace Assistant (iWA) in the Modular Production System (top view). Green indicates reachable components and orange is not reachable.

acts as the human-in-the-loop interface to close the gap between the simulated workspace design and the real-world. The collaborative robot and its collaborative functions like the kinesthetic freedrive mode are utilized as an interactive tool for transferring the virtual layout into the real world setup and vise versa.

This approach is based on methods of Modular Production Systems such as shown in Fig. 1. Modular and composable hardware components are used to divide fabrication properties in manageable chunks with predefined duties. However, the operator has to consider the position and orientation of each component, the connection between components according to the assembly process, and the reachability of working points to ensure a robot solution with short paths to achieve efficient cycle times (cf. Fig. 2). These requirements challenge the operator in finding an appropriate workspace layout. The exploration and layout of the robot workspace uses a procedural algorithm inspired and known from game development, where 2D and 3D levels are auto-generated. The algorithm places the modular components automatically in robot workspaces, according to a defined strategy, and refines the layout while considering the decisions of the operator.

In the following Section II, we present the related work about smart factories, simulation-based approaches, and recent industrial developments. Section III describes the main concept of this contribution and details about implementation are explained in Section IV. Our evaluation is presented in Section V before concluding in Section VI.

II. Related Work

Intelligent systems, data-driven developments, and mass-customized productions push the frontiers of conventional manufacturing toward Smart Factories, jointly realizing the leading ideas of Industry 4.0. A rich and intensive simulation phase reduces drastically costs, time and efforts during the integration phase [3]. Our proposed system indents a seamless integration into the existing development process of SMEs using available in-house data.

The generation and exploration of a workspace layout for robotic applications is related to the analysis about the ability to reach objects with the end-effector and to manipulate these objects within the workspace. A quantitative measure for manipulability has been introduced by Yoshikawa [2] and represents the ability to position and orientate the robot's end-effector. The goal is to achieve the best postures for a given task to increase the manipulating ability of the robot arm. This ability is expressed by the manipulability index, represented by an ellipsoid at the end-effector, which is the three-dimensional, volumetric equivalence of an ellipse. The ellipsoid approach of Yoshikawa was further developed by Vahrenkamp and takes robot specific constraints into account, such as joint limits or distances between links and parts of the robot [4]. This approach delivers a more accurate quantification on the robot agility considering the selection of applicable robot postures in terms of grasping and manipulation.

Automated workspace planning is further related to the search for the initial robot position within the workspace and the ability to reach objects with the gripper. Lueth et al. have presented an automated layout planning approach with the goal to eliminate human intervention and "check-and-change" correction loops by a fully automated process [5]. This approach checks the workspace for valid and reachable working points to generate collision free robot paths. An algorithmic coverage for the search of the initial position combined with the reachability is represented by the approaches from Zacharias et al. [6]. The authors have shown how the workspace of the robot is captured in a structured way and visualized in a reachability graph to analyze the manipulability and reachability of objects in this space. Exemplary implementations were published in open-source libraries such as *Reuleaux* [7] and *REACH* [8].

Smart Factories require intelligent tools for seamless integration of collaborative robots, sensors and processes for testing and evaluating the setup. Recent developments in the industrial sector have released products like the *Mikado ARC* system for bin-picking applications [9]. This software tool offers an embedded offline simulator accompanied by a structured and guided process to test bin-picking, including the gripper and the peripheral equipment. This allows a manual definition of multiple grasping poses, using the so called *Grasp Inspector*. The integrated path planner searches for valid motion solutions to the stored grasp poses of objects identified in the real bin. An open-source and research-oriented counter-part to this system, is represented by the *MoveIt! Task Constructor* used for hierarchical organization of basic stages concerning motion and grasp planning [10]. Moreover, the product of *ArtiMinds*, a commercial software-suite for robot programming, offers a tool for the manipulability calculation while a part is placed in a robot workcell [11], [12]. Visual and color-coded feedback informs the operator if a part is reachable and manipulable before the robot motion is simulated or transferred on the real robot.

Research on active, dialog-based assistance has been performed by Racca et al. [13]. The authors present in their work an approach for intuitive robot programming based on predefined user dialogs to tune motion parameters manually and in cooperation with the operator. This learning process collects user feedback in the form of directional answers, such as “higher,” “lower,” or “fine,” to adjust interactively robot velocities and forces.

This contribution is based on our previous work[1] combining industrial aspects with active operator assistance in a Modular Production System (MPS) using a collaborative robot for material handling [14], [15]. Further, the co-bot is facilitated for the rapid reconfiguration of the modular workspace. An integrated simulation, representing the robot and workcell digitally, supports the operator during the dynamic integration of components (cf. Fig. 1). In contrast to the algorithmic approaches mentioned above, this contribution takes advantage of the operator’s cognitive skills to initialize the automatic layout algorithm and enhance the solution search for a valid workspace layout.

III. Concept

The main concept is a hybrid approach comprised of on- and offline robot programming methods, where the operator is guided and supported through an Interactive Workspace Assistant (iWA). Dedicated operator hands-on stages are combined with automated computation and simulation stages for an auto-generation of robotic workspace layouts with composable and modular components (see Fig. 3). The idea behind the modular system is that each component, utilized and specialized for a manufacturing duty, is equipped with a self-description. This self-description contains part-specific details such as the reference frames to represent robot working points within this component, volumetric 3D-models for visual operations with the simulation, and, of course, a description about the manufacturing functionalities and capabilities. Once a component is used in a MPS, the self-description is invoked as soon as the component is initialized by the production system or parameterized for the manufacturing.

Modularity is a modern and effective method to increase flexibility, which is highly demanded today in the production. However, modularity increases the weight, footprint and space of each component. The space is limited in shopfloor environments, as well as the accessibility of robots. Consequently, a co-bot with a limited operational area must serve and handle as many components as possible in a highly dense workspace.

The challenge for the operator is to place components with different, unregular shapes close to each other. At the same time, the operator must consider the sequence of the assembly process and the manipulability of working points to ensure a robot solution with short movements to achieve efficient cycle times. Considering and optimizing all these requirements challenge the operator in finding an appropriate workspace layout.

Our approach emphasizes the algorithmic exploration of the composable workspace in the simulation before moving toward execution and application on the real robot. Exploring the environment in simulation shows analogies to the automatic generation of 3D environments in the gaming industry. A commonly used technique, known as procedural design, breaks down a game level in fundamental elements and interrelated structures [16], which are combined algorithmically into a valid and purposeful environment. This technique is adopted for a gradual workspace layout of the robot environment. The prerequisite for such a gradual workspace layout are composable and reusable building blocks, represented in our approach, by the modular and flexible components ready to be plugged-in and used for the manufacturing in the MPS.

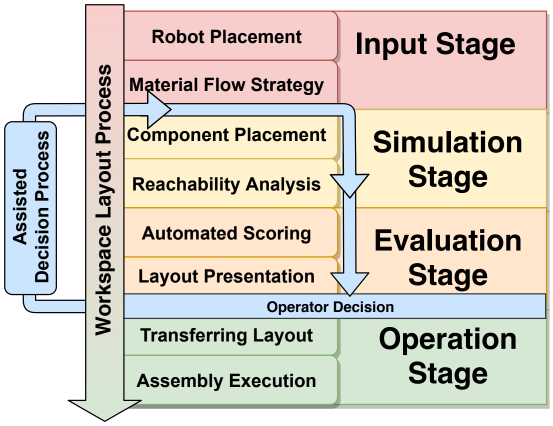


Fig. 3: Workflow chart for the automatic layout planning containing the stages from input to the hands-on operation stage.

A. Modular Components for a gradual Workspace Layout

Modular Production Systems use flexible and modular components, where the material flow in the workspace is handled by a collaborative robot [17]. A standardized mechanical format allows the components to be fixed at any point in the workspace. Defined calibration points ensure a quick integration using the robot’s freedrive and kinesthetic functions to localize its positions. Further the components can be connected with and registered to the production system due to a standardized electrical and data structure.

This quick and straightforward integration of components in a robot-based automation system is known as Plug-and-Produce [18]–[21]. Once a component is plugged-in and connected to production system, the self-description of the hardware component is transmitted to the production system. The self-description is modeled according to the specification of the Automation Markup Language (AML) to instantiate a component at run-time with defined functionality and communication interfaces.

Concerning mandatory robot functionalities, the self-description contains references to CAD models used for visual aspects of the 3D representation in the simulation and internal computation of collision free movements with the path planner. A frame provider organizes the working points for the robot and updates the transformation tree of dynamically placed and plugged-in components during the workspace exploration, initial setup or reconfiguration of the production system.

[1] User Study Video: https://youtu.be/Mf5iPpiD2NE

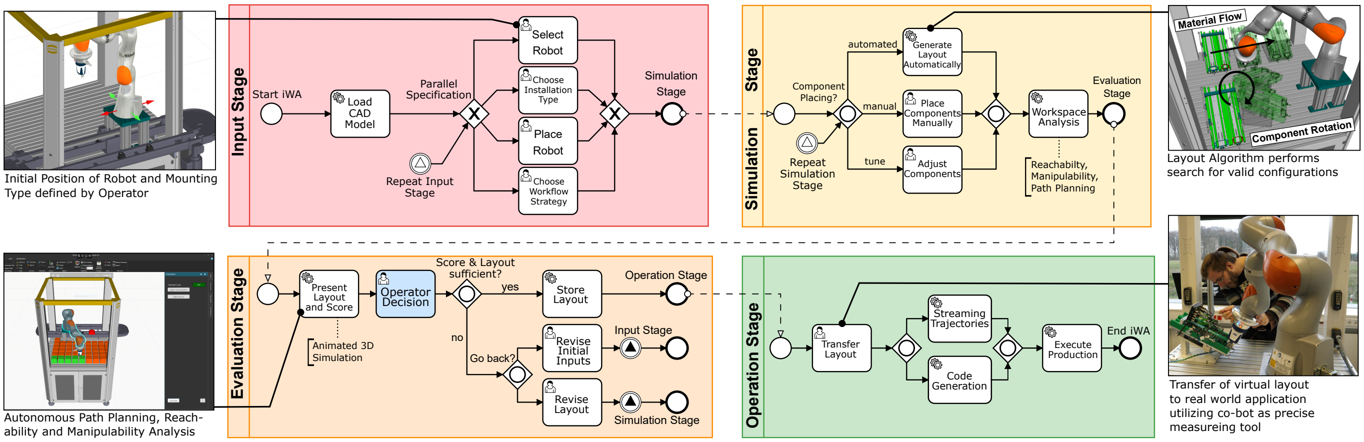


Fig. 4: Simulation Model of the iWA System combining technical functions and operator activities in a shared model of BPMN2.0 notation.

A detailed description on the Modular Production System and the Plug-and-Produce process for an automated the robot-based production was published in our previous work [15].

B. Workflow for an interactive Workspace Planning

The workflow for generating a workspace layout interactively is divided into four stages (cf. Fig. 3). In the Input Stage (1), the operator defines the initial robot position, specifies the material flow strategy and chooses process parameters needed for the assembly. Appropriate material flow strategies are depending on the manufacturing task and set by the operator. Valid strategies are, for example, linear ones to achieve a directed flow through the workcell, clockwise, counterclockwise or U-shaped to follow the lean production workflow, or a greedy strategy that places prioritized components before others at well reachable positions.

Once the initial inputs are declared, the automatized Simulation Stage (2) performs a search for a valid workspace composition. The procedural algorithm, presented in Sec. III-D and inspired by techniques in game level design, is used for positioning the modular components. The result of the automated planning is presented to the operator in the Evaluation Stage (3) and the workspace layout is automatically rated by a score. The score is based on coverage of the workspace according to the robot's footprint, reachability and manipulability of components. The operator decides, whether the workspace layout and score are sufficient, and transfers the computed layout to the real world during the Operation Stage (4).

In case of an unsatisfied score or workspace layout, the operator returns to Stage (1) and revises the specifications such as the robot placement or material flow strategy. The revision is advised according to the simulated result, e.g. placing the robot or component more advantageously in the workspace.

C. The Model behind the Operator Activities

It is crucial to model the human activities, interactions, and behaviors besides the functional system capabilities to obtain a successful simulation tool, which cooperates closely with the human. For this reason, the presented iWA system uses a process model in the simulation back-end combining both in the BPMN2.0 modeling language [22] shown in Fig. 4.

The process model follows the sequential workflow automatically from the start to the end of workspace assistant through the previously described workflow stages. The process stops at *User Tasks*, indicated by a human icon inside the process activity, and requires an action performed by a human operator. Such actions in the iWA system are placing the robot, choosing the desired workflow strategy, or finalizing all conditions of a parallel execution to continue the process and to get in the next stage.

Automated or scripted processes, like the generation of the workspace layout or performing the reachability analysis, are invoked by *Service Tasks* (gearbox icon). A service task doesn't require any operator interaction and continues the sequence flow of the process once it's finished.

A central part of the iWA model are operator decisions that require human interactions with the components in the simulation or the assembly process. Such decisions are denoted by *Gateways*, which compare the decision with the condition on the outgoing sequence flow. A decision can be made whether a component should be placed manually or automatically in the layout. *Signal Events* allow jumps from the current process stage into a particular stage and are indicated in the BPMN model by the triangle. For instance, it is possible to jump back from the evaluation to the simulation stage performing a fine-tuning and adjusting of single components. This allows an active influence of the operator on the workspace layout.

A following human-in-the-loop process (cf. *Operation Stage*) intends to close the simulation-gap between the simulated environment and the real workspace. However, the iWA system must perform an automated and vendor-specific code generation or joint angle streaming to supply the desired target platform with the robot motion and behavior information. Either the operator transfers the base coordinates of the components manually into the real world, or the operator uses calibration methods to exploit collaborative and kinematic functionalities of the robot such as the approach described in our previous publication [14].

Algorithm 1 Excerpt of Automatic Workspace Planning

Input: Workspace, Component cmp, Environment e, MaterialFlow

Output: WorkspaceLayout, Reachability, Manipulability, Path

```
while $Workspace \neg explored$ do
    $placeComponent(Workspace)$
    if $cmp = placed$ then
        $reachability \leftarrow computeInvKin$
        if $InvKinSolution\ found$ then
            $path \leftarrow computeCollisonFreePath(cmp, e)$
            $mnpIdx \leftarrow computeManipulability$
            $store(reachability,\ mnpIdx,\ path)$
    if $cmp = inCollision\ \&\ rotation < 360°$ then
        $rotate(cmp)$
        return
    else $shiftComponent(MaterialFlow)$
return $WorkspaceLayout \leftarrow cmp$
```

D. Algorithm for automatic Workspace Layout

The planning algorithm for workspace layout organizes and places composable modular components during the automated planning sequence. An excerpt of the algorithm is presented in Alg. 1, which starts with the coarse exploration of the workspace and continues with component placement inside the working area of the robot. The component is moved along a path of the predefined material flow, then rotated locally, and tested for reachability of working points applying inverse kinematics. In case of a successful solution, the collision free path through the workspace is computed, which includes the avoidance of obstacles like dynamically placed components and the static environment of the manufacturing cell.

It is essential in industrial assembly and production scenarios to reach the target with the robot end-effector in a specific goal pose and manipulate the object precisely around this working point. For example, an object from a material storage must be pulled out on a linear path and held in a specific orientation during the manipulation process. Solving such a manipulation problem is non-trivial task, especially in robot configurations near the joint limits. Thus, this algorithm includes the indexing of the ability to manipulate an object in the current working point (manipulability index). The manipulability index, introduced by Yoshikawa [2], was used for this contribution and extended by a penalization function to punish joint configurations near the upper and lower joint limits according to Tsai [23]. The workspace exploration has reached the termination condition when the components are no more shiftable and rotatable without self- or environment collision. The gathered information about the reachability and manipulability form the basis of optimizing the workspace, which is evaluated in Sec. V.

IV. Implementation of the System Architecture

The following section provides details about the software architecture, levels of robot and task programming, and the according user roles operating with the system, followed by the possible deployment strategies. The robot simulation software used for this implementation is Visual Components [24]. The extension points of the simulation software allow integrating a low-level robot control for the path planning and time-based motion execution. The intermediate-level implements composable and hierarchically organized robotic skills. Whereas the high-level is used for task programming and extension of the user interface for interactive and user-oriented features of the iWA system.

A. Roles in the robotic software development

Moving toward digital factories and Industry 4.0 applications, a user-centered approach is a mandatory prerequisite for embedding a simulation tool in the development and engineering process [25], [26]. This approach and concept uses a digital human representation to consider the human capabilities and needs, as shown in Fig. 6. This basic model follows the system theory, where the operator gives input to the simulation in form of perception and knowledge. The simulation transforms through a computer-aided process the user inputs into production-related outputs (e.g. robot trajectories). Before the output goal is reached, usually several iterations of a feedback loop are needed. After receiving the input, the operator decides if the result is going to be accepted, manipulated, optimized, or rejected.

During the development of applications for industrial duties, various persons are involved before a turnkey system is handed over to the customer. The development process of a simulation-based environment reveals three leading roles in robot programming: *Expert-Users* for low-level back-end programming, *Intermediate-Users* for interfacing between the low-level and the application, and the *End-Users* for applying an industrial task prepared in simulation and for shopfloor execution.

The iWA focuses strongly on the requirements of the *End-Users*. These application engineers operate on the shop floor under success and time pressure. It is excepted that *End-Users* have broad knowledge about the product and the accompanying production process. However, these engineers are not necessarily robotic experts. For this reason, an efficient and supportive tool is mandatory that encapsulates the complexity

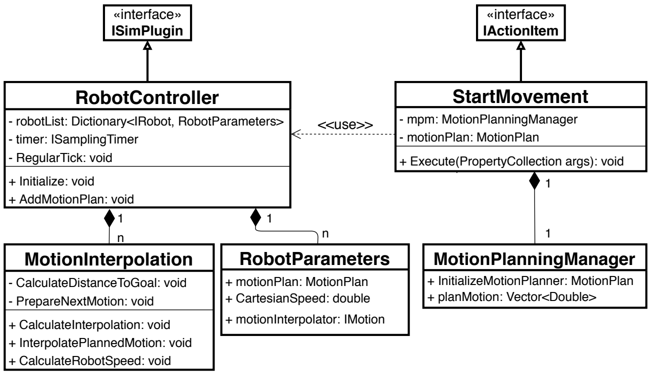

«interface»
ISimPlugin
«interface»
IActionItem
RobotController
- robotList: Dictionary<IRobot, RobotParameters>
- timer: ISamplingTimer
- RegularTick: void
+ Initialize: void
+ AddMotionPlan: void
<<use>>
StartMovement
- mpm: MotionPlanningManager
- motionPlan: MotionPlan
+ Execute(PropertyCollection args): void
1
n
1
n
1
1
MotionInterpolation
- CalculateDistanceToGoal: void
- PrepareNextMotion: void
+ CalculateInterpolation: void
+ InterpolatePlannedMotion: void
+ CalculateRobotSpeed: void
RobotParameters
+ motionPlan: MotionPlan
+ CartesianSpeed: double
+ motionInterpolator: IMotion
MotionPlanningManager
+ InitializeMotionPlanner: MotionPlan
+ planMotion: Vector<Double>

Fig. 5: Class diagram of the iWA simulation back-end.

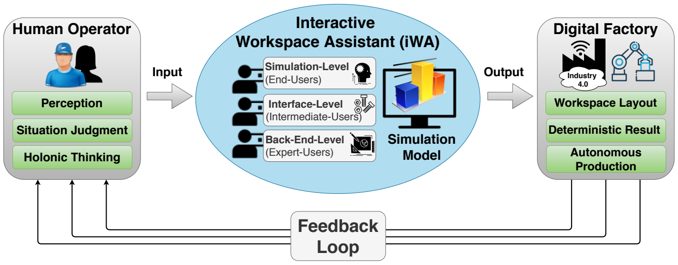


Fig. 6: Implementation prototype representing the model of the operator including a feedback loop to influence and optimize simulation results.

of robotic applications in easy-to-use functions to complete the demanded task of the client and to satisfy the stakeholders.

A separation of tasks is mandatory to achieve a high specialization and efficiency on each programming level. The *End-User* is responsible for the successful deployment of the software used at run-time, establishing data exchange between system components, and ensure a correct task execution. The link between the application engineers and the low-level are *Intermediate-Users* that combine and wrap the provided low-level functions into useful and task-oriented function blocks and robotic skills. Robotic skills encapsulate the complexity of a specific robot, its characteristic features, and programming language to allow a flexible task adjustment through external parameterization of function blocks [27]. The *Expert-Users* program the low-level simulation back-end. These users take care of the controller that moves the robot, implement path planners for trajectory generation and collision avoidance, and program algorithms that smooth robot trajectories to reduce vibrations. Because this programming level requires highly-skilled expertise, an application engineer is not involved in the development process on this level. The iWA system is represented on all user levels. However, the application engineer operates only on the *End-User* level with the functions provided by the graphical user interface and the three-dimensional robot simulation.

B. Data structure of the simulation back-end

The data structure is shown in the UML Class Diagram of Fig. 5. The *IActionItem* provides access from the high-level Python programming API to the low-level infrastructure, which is used by *Intermediate-Users*. For a convenient operating with the robot, we have developed the *MotionPlanningManager* to organize the motion requests from one or multiple robots in the simulated scene. At each simulation-run, the robots are instantiated and the 3D scene is passed to perform path planning and collision avoidance on the current simulation layout. Further, the process plan and task sequence is modeled and defined by a set of ordered task frames. The position and orientation of the task frames can be adjusted by the operator or layout algorithm before each simulation run. The appropriate motion request between a pair of frames is manged at run-time by the *MotionPlanningManager* for each partial movement of a robot trajectory.

The interface class of the *ISimPlugin* is used to get full access to the simulation scene, the properties of a component and the user-interface. An instance of the *RobotController* is used for a sequential partition of the simulation time to update the robot's joint angles at each timestep. The *MotionInterpolator* slices a trajectory into time-based steps, that are executed by the *RobotController*. Robot specific parameters such as the 3D-model of the robotic manipulator, upper and lower joint limits, or the robot position in the simulated world are stored and used from the data structure of the *RobotParameter* class.

C. Deployment

The deployment strategy of the iWA system is a crucial part to span the arc from virtual engineering to the real application. One solution to close this simulation gap is the automatic code generation for the individual parts and subsystems of the modular manufacturing system. This includes generation of code by a parameterizable post-processor based on the virtual model for the behavior of the robot, functionalities and ready-to-use interfaces of the modular components, and code generation for the coordination of the process control unit (cf. *Operation Stage* in Fig. 4).

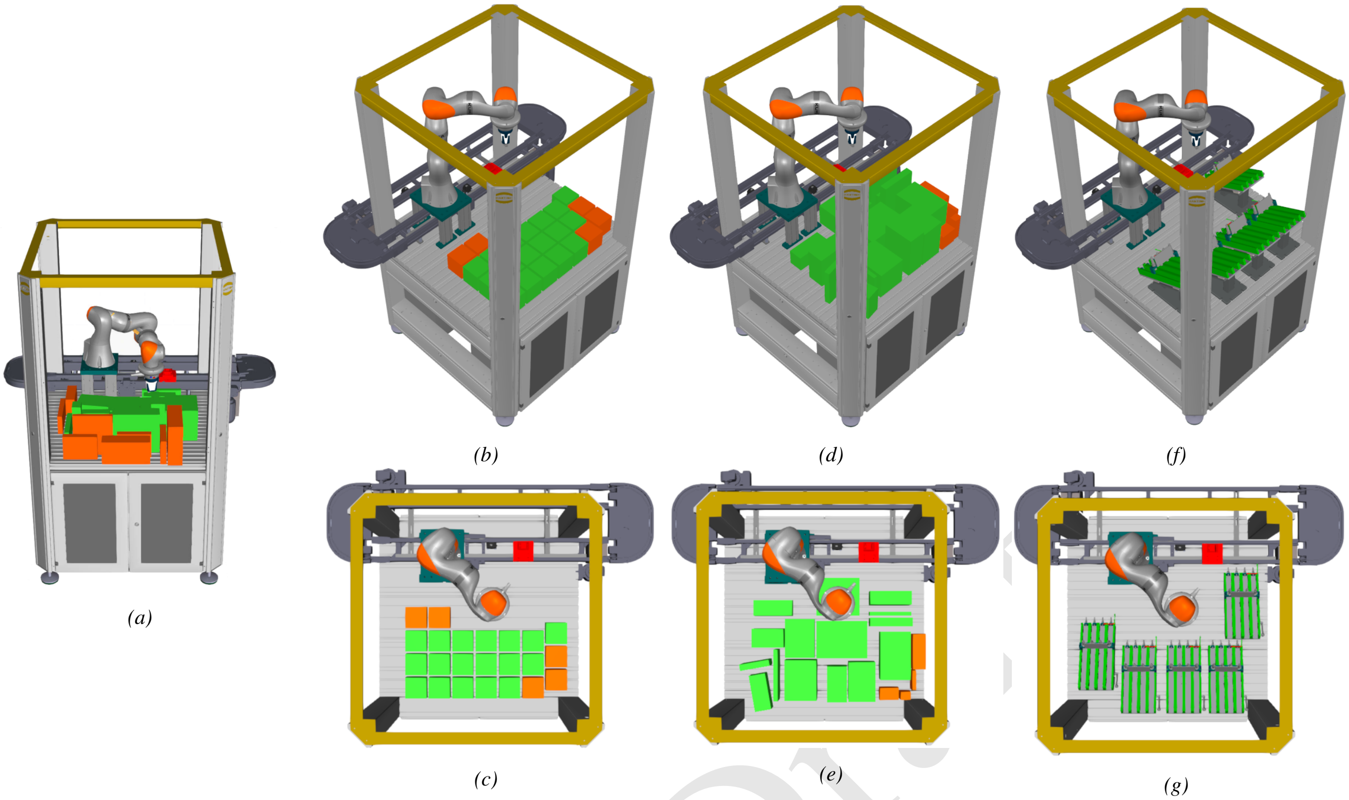
Another suitable approach for exchanging states between these both worlds is the streaming of robot joint angles and condition monitoring of relevant parameters to adopt these properties into the virtual simulation model. The capabilities of the robot, such as the freedrive or kinesthetic function, are practical methods to utilize and benefit from the collaborative technology of the robot. Further, it is possible to use calibration tools attached to the robot gripper to pass the position and orientation of components via specified calibration points from the real world to the simulation model.

The human operator is intended to be the essential link between the virtual and real world of the iWA system. In the simplest case, the operator applies and deploys the computed layout by transferring coordinates and rotation angles of components into the coordinate system of the workspace. Using a metering rule and angle meter is an approach of low technical complexity, however it presumes more background knowledge of the operator and leaves space for errors.

In a technically more advanced system, the collaborative robot indicates with the gripper-tip, how and where components are meant to be placed. Alternatively, a vision system may be used for recognition of components in the workspace. Oriented QR-Codes (Quick Response) support as visual markers the localization process. Such an approach was presented by Wein et al. embedded in a robotic system for industrial assemblies [28]. Once the positions are stored and represented in the simulation model, the collision free paths have to be computed by the path planner before being streamed or executed on the real robot.

V. Evaluation

The evaluation of the Interactive Workspace Assistant was carried out to analyze the initial capabilities of the system. The experiments were conducted on a collaborative robot with



(a) (b) (c) (d) (e) (f) (g)

Fig. 7: (a) Grasping pose during the iterative workspace exploration. (b-c) Workspace exploration with same sized components, and linear material flow strategy. (d-e) Components with varying size, rotated during exploration, placed greedy around central point in the workspace. (f-g) storage component for simulating a real world task in a robotic workcell, placing followed a spiral strategy.

7 degrees of freedom (Kuka IIWA 7 R800) in the constrained workspace of a Modular Production System presented in Fig. 1 and published in [14]. In this evaluation, the focus is on the simulation and generation of the workspace layout to validate the functionality of the automated and algorithmic planning. A valid workspace Lueth defined as an area where all components are reachable, the robot is not blocked by the static environment, and the paths between task frames are collision free and suitable for the application [5].

Eq. 1 denotes the policy for calculating the score S_{iWA} used for the evaluation of the layout. Let A_{ws} be the total area of the available workspace and A_{cmp} the accumulated area, respectively the footprint, of the modular components placed in the robotic cell. The partial fraction is called *Package Density* and represents the arithmetic mean and the efficiency ratio of the occupied workspace.

$$S_{iWA} = \frac{1}{3} \cdot \left(\frac{\sum A_{cmp}}{A_{ws}} + \frac{\sum R_{cmp}}{\sum Cmp} + \frac{\sum M'_{Idx}}{\sum F_{cmp}} \right) \quad (1)$$

Further, the score includes the quotient of reached components R_{cmp} in relation to the total number of modular components Cmp representing the reachability ratio. The manipulability M_{Idx} defines the ability to move and manipulate a grasped object around a task frame F_{cmp}, and M'_{Idx} normalizes the manipulability range. The implementation follows Zhang et al. to calculate the manipulability index using the screw theory [29]. The experiments were conducted with generic, but varying, components of low shape complexity to achieve a basic understanding of the presented method. Further, a complex-shaped storage component was used to simulate a real world assembly application. The objectives for workspace exploration have been set as followed:

1) Same sized components containing six task frames, and a linear material flow strategy.
2) Components with randomized size, rotated during linear exploration to increase package density, placed greedy around workcell center.
3) Storage component containing four task frames, following a spiral placement strategy.

The results of the workspace exploration are depicted in Fig. 7 and relevant parameters are summarized in Tab. I. If the reachability of a task frame has been tested successfully and a collision free path was computed, the components are colored green otherwise orange. Objective (1) shows a decent performance reaching 108 out of 138 task frames, respectively, 18 components were placed successfully after the initial run. However, the resulting score of 35.69% indicates room for improvement, and the operator must increase the reachability in the next iteration by rearranging the unreachable components. The manipulability envelope has an impact on the score. The cubic components of low height are placed in areas where a low manipulability dominates due to the robot's hardware design. The generated layout of Objective (2) shows a higher coverage of reached components, a denser occluded workspace area, and on average the components were generated with greater height and placed in zones of higher manipulability. Objective (3) outperforms the previous layouts using a spiral placement strategy for the storage components (50.16%). But the top score can't be traced back to the placement strategy only. It is justified by a combination of workspace coverage, manipulability, and strategy. A future investigation must examine a weighting on the partial fractions to fine-tune the equation (Eq. 1) and to increase the precision of quantification and derivation of context-related advice for the operator.

TABLE I: Results of the initial Evaluation for the Interactive Workspace Assistant.

Obj.	**Cmp**[1]	R_{cmp}[2]	A_{cmp}[3]	F_{cmp}[4]	**Score**
1	23	18	276768	108	35.69%
2	19	16	384166	96	40.19%
3	5	5	484800	20	50.16%

[1] *Number of Components* placed in Workspace.
[2] *Reachabilty of Components* in Workspace.
[3] *Footprint of reached Cmp in* [mm^2], $A_{ws} = 960400mm^2$.
[4] *Total amount of Reachable Task Frames*.

VI. Conclusion

This contribution has shown a conceptual approach taking into account the experience-based user interaction to address the problem of the automatic workspace layout, where the workspace is gradually composed, optimized, and revised by the operator's experience. Future work focuses on an extensive user study to gain more insights about the acceptance, usability, and needs of operators working with the proposed assistant. The evaluation has shown initial results about the operational mode, which will be used in future work for the assessment of the current situation and the derivation of assistant strategies to support the operator.

References

[1] *Industrial Robots: Robot Investment Reaches Record 16.5 billion USD*, (accessed on December 09, 2019), https://ifr.org/ifr-press-releases/news/robot-investment-reaches-record-16.5-billion-usd.

[2] T. Yoshikawa, "Manipulability of Robotic Mechanisms." *International Journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[3] VDI-Fachbereich Fabrikplanung und Betrieb, *Digital factory Digital Factory Operations Part 2*. VDI Gesellschaft Produktion und Logistik, 2011.

[4] N. Vahrenkamp and T. Asfour, "Representing the robot's workspace through constrained manipulability analysis," *Autonomous Robots*, vol. 38, no. 1, pp. 17–30, 2015.

[5] T. C. Lueth, "Automated planning of robot workcell layouts," in *Proceedings 1992 IEEE International Conference on Robotics and Automation*. IEEE, 1992, pp. 1103–1108.

[6] F. Zacharias, C. Borst, and G. Hirzinger, "Capturing robot workspace structure: representing robot capabilities," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Ieee, 2007, pp. 3229–3236.

[7] A. Makhal and A. K. Goins, "Reuleaux: Robot base placement by reachability analysis," in *2018 Second IEEE International Conference on Robotic Computing (IRC)*. IEEE, 2018, pp. 137–142.

[8] M. Ripperger, *REACH - Robotic Evaluation And Comparison Heuristic*, (accessed on December 20, 2019), https://github.com/ros-industrial/reach.

[9] *Mikado ARC - Adaptive Robot Control. A Programming Suite for Bin-Picking Tasks*, (accessed on March 23, 2020), https://de.ids-imaging.com/mikado.htmln.

[10] M. Görner, R. Haschke, H. Ritter, and J. Zhang, "MoveIt! Task Constructor for task-level motion planning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 190–196.

[11] R. Jaekel and G. Dirschl, "Method and system for programming a robot," U.S. Patent 2017/0 190 052A1, July 6, 2017.

[12] *ArtiMinds Robotics: Robot Cell Design via Manipulability Calculation*, (accessed on July 13, 2020), https://youtu.be/hXFVvjVp6cA.

[13] M. Racca, V. Kyrki, and M. Cakmak, "Interactive Tuning of Robot Program Parameters via Expected Divergence Maximization," pp. 629–638, 2020.

[14] M. Wojtynek, J. J. Steil, and S. Wrede, "Plug, Plan and Produce as Enabler for Easy Workcell Setup and Collaborative Robot Programming in Smart Factories," *KI - Künstliche Intelligenz*, May 2019.

[15] M. Wojtynek, H. Oestreich, O. Beyer, and S. Wrede, "Collaborative and Robot-Based Plug & Produce for Rapid Reconfiguration of Modular Production Systems," in *International Symposium on System Integration*. IEEE, 2017.

[16] K. Compton and M. Mateas, "Procedural level design for platform games." in *AIIDE*, 2006, pp. 109–111.

[17] R. Bokranz and K. Landau, *Handbuch Industrial Engineering: Produktivitätsmanagement mit MTM*, 2nd ed. Stuttgart: Schäffer-Poeschel, 2012.

[18] G. Reinhart, S. Krug, S. Hüttner, Z. Mari, F. Riedelbauch, and M. Schlögel, "Automatic configuration (plug & produce) of industrial ethernet networks," in *2010 9th IEEE/IAS International Conference on Industry Applications-INDUSCON 2010*. IEEE, 2010, pp. 1–6.

[19] S. Schriegel and L. Wisniewski, "Investigation in automatic determination of time synchronization accuracy of PTP networks with the objective of Plug-and-Work," in *2014 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*, Sept 2014, pp. 20–25.

[20] M. Schleipen, A. Lüder, O. Sauer, H. Flatt, and J. Jasperneite, "Requirements and concept for Plug-and-Work," *at – Automatisierungstechnik*, pp. 801–820, 2015.

[21] J. Pfrommer, D. Stogl, K. Aleksandrov, S. E. Navarro, B. Hein, and J. Beyerer, "Plug & produce by modelling skills and service-oriented orchestration of reconfigurable manufacturing systems," *At-Automatisierungstechnik*, vol. 63, no. 10, pp. 790–800, 2015.

[22] *Business Process Model and Notation (BPMN2.0) - Standard for Business Processes diagrams.*, (accessed on April 03, 2020), https://www.omg.org/spec/BPMN.

[23] M.-J. Tsai, "Workspace geometric characterization and manipulability of industrial robots," Ph.D. dissertation, The Ohio State University, 1986.

[24] *Visual Components – Simulation Software for Robots in Manufacturing Environments.*, (accessed on March 20, 2020), www.visualcomponents.com.

[25] C. Wittenberg, "Human-CPS Interaction-Requirements and human-machine interaction methods for the Industry 4.0," *IFAC-PapersOnLine*, vol. 49, no. 19, pp. 420–425, 2016.

[26] C. Prinz, F. Morlock, S. Freith, N. Kreggenfeld, D. Kreimeier, and B. Kuhlenkötter, "Learning factory modules for smart factories in Industrie 4.0," *Procedia CiRp*, vol. 54, pp. 113–118, 2016.

[27] S. Bøgh, O. S. Nielsen, M. R. Pedersen, V. Krüger, and O. Madsen, "Does your robot have skills?" in *Proceedings of the 43rd International Symposium on Robotics*, vol. 6. Verlag, 2012.

[28] S. Wein, L. Wolff, A. Malik, S. Storms, and W. Herfs, "Concept for Automated Robot Programming Using Image Processing," in *Congress of the German Academic Association for Production Technology*. Springer, 2018, pp. 705–714.

[29] L. Zhang, S. Guo, Y. Huang, and X. Xiong, "Kinematic singularity analysis and simulation for dof anthropomorphic manipulator."