Robotics and Computer Integrated Manufacturing 66 (2020) 101979

Contents lists available at ScienceDirect

Robotics and Computer Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Smart hardware integration with advanced robot programming technologies for efficient reconfiguration of robot workcells

Timotej Gašpar[*,a], Miha Deniša[a], Primož Radanovič[a], Barry Ridge[a], T. Rajeeth Savarimuthu[b], Aljaž Kramberger[b,a], Marc Priggemeyer[c], Jürgen Roßmann[c], Florentin Wörgötter[d], Tatyana Ivanovska[d], Shahab Parizi[e], Žiga Gosar[f], Igor Kovač[a], Aleš Ude[a,g]

[a] Humanoid and Cognitive Robotics Lab, Department of Automatics, Biocybernetics and Robotics, Jožef Stefan Institute Jamova cesta 39, 1000 Ljubljana, Slovenia
[b] Mærsk McKinney Møller Institute, University of Southern Denmark, 5230, Odense M, Denmark
[c] Institute for Man-Machine Interaction, RWTH Aachen University, Ahornstraße 55, 52074 Aachen, Germany
[d] Bernstein Center for Computational Neuroscience, Third Physics Institute, Georg-August-Universität Göttingen, Friedrich-Hund-Platz 1, 37077 Göttingen, Germany
[e] Blue Ocean Robotics, Niels Bohrs Allé 185, 5220 Odense SØ, Denmark
[f] Elvez d.o.o., Ulica Antona Tomšiča 35, 1294 Višnja Gora, Slovenia
[g] Faculty of Electrical Engineering, University of Ljubljana, Tržaška cesta 25, 1000 Ljubljana, Slovenia

ARTICLE INFO

Keywords:
Reconfigurable manufacturing systems
Passive reconfigurable hardware
ROS
Programming by demonstration
Industry 4.0

ABSTRACT

The manufacturing industry is seeing an increase in demand for more custom-made, low-volume production. This type of production is rarely automated and is to a large extent still performed manually. To keep up with the competition and market demands, manufacturers will have to undertake the effort to automate such manufacturing processes. However, automating low-volume production is no small feat as the solution should be adaptable and future proof to unexpected changes in customers' demands. In this paper, we propose a reconfigurable robot workcell aimed at automating low-volume production. The developed workcell can adapt to the changes in manufacturing processes by employing a number of passive, reconfigurable hardware elements, supported by the ROS-based, modular control software. To further facilitate and expedite the setup process, we integrated intuitive, user-friendly robot programming methods with the available hardware. The system was evaluated by implementing five production processes from different manufacturing industries.

1. Introduction

In a diverse global market, the ability to adapt to increasingly frequent changes in clients' demands has become crucial to maintain competitiveness. The increasing demand for higher production flexibility and lower production costs has been pushing the development towards more adaptable automation solutions. The manufacturing sector had to evolve to cope with these changes. New approaches have been developed, ranging from Flexible Manufacturing Systems (FMS) to the logical next step, Reconfigurable Manufacturing Systems (RMS). The RMS paradigm advocates manufacturing systems that can rapidly and efficiently adjust production capacity and functionality to meet sudden changes in market demands [1].

By enabling physical interaction with humans, collaborative robots introduce new skill and task-level robot programming methods. These new approaches make robots a viable part of the array of reconfigurable modules within an RMS. However, to achieve the desired RMS-like performance, all the hardware and software components within the system should be modular by design [2].

The objective of the work presented in this paper was to design and implement an RMS in shape of a reconfigurable robot workcell, suitable not only for large production lines but also for low-volume high-diversity production. The proposed workcell consists of the following crucial components to achieve the stated goal:

- Modular and largely passive hardware components aimed towards cost-effective, autonomous workcell reconfiguration.
- A software backbone based on ROS, which provides modularity of

* Corresponding author.
E-mail addresses: timotej.gaspar@ijs.si (T. Gašpar), miha.denisa@ijs.si (M. Deniša), primoz.radanovic@ijs.si (P. Radanovič), barry.ridge@ijs.si (B. Ridge), trs@mmmi.sdu.dk (T.R. Savarimuthu), alk@mmmi.sdu.dk (A. Kramberger), priggemeyer@mmi.rwth-aachen.de (M. Priggemeyer), rossmann@mmi.rwth-aachen.de (J. Roßmann), worgott@gwdg.de (F. Wörgötter), tiva@uni-goettingen.de (T. Ivanovska), msp@blue-ocean-robotics.com (S. Parizi), ziga.gosar@elvez.si (Ž. Gosar), igor.kovac@ijs.si (I. Kovač), ales.ude@ijs.si (A. Ude).

https://doi.org/10.1016/j.rcim.2020.101979
Received 11 October 2019; Received in revised form 18 March 2020; Accepted 22 March 2020
Available online 19 May 2020

all software components.
- Fast programming of assembly tasks through kinesthetic guidance and script-based state machine description methods supported by visual programming interfaces.
- Support for quick workcell reconfiguration by manipulating and exchanging the available hardware components, facilitated by a digital twin.

The proposed approaches were evaluated through the implementation of multiple, real-life industrial production processes involving automated robot assembly.

In the following we first present the state of the art. In Section 3 we describe the hardware design aspects of the developed reconfigurable workcell. The software backbone of the cell is discussed in Section 4, while the programming technologies for fast robot programming are presented in Section 5. We present the aspects of reconfiguration of the proposed cell in Section 6 and the methodologies applied for fast workcell setup in Section 7. The results of the evaluation, obtained by implementing the real industrial use cases, are presented in Section 8.

2. Literature review

Since its introduction, the Reconfigurable Manufacturing Systems (RMS) paradigm gained significant traction within the research community [3–5]. The key characteristics of such systems are: *customization, convertibility, scalability, modularity, integrability* and *diagnosability* [6]. However, due to the increased complexity of design and lower throughput of RMS, compared to more dedicated systems, it is important to thoroughly evaluate if such a manufacturing model is appropriate for each specific production process [7]. RMS bring added value in manufacturing processes where changes in production happen relatively often [8]. Low-volume and make-to-order manufacturing [9] is in high proportion coming from Small and Medium Enterprises (SMEs) and suffers from delays emerging from process planning [10,11]. Brunoe et al. [12] argue that SMEs benefit most by implementing the RMS paradigm in their production at the workstation level, rather than at the production line level. This has been implemented to a degree by Makris et al. [13] who developed a flexible dual-robot system with a human collaborator. They integrated their robot into a virtual twin in order to efficiently organize all production entities [14]. But in order to make RMS more viable for SMEs, it is important to tackle the issue of the high investment costs when applying such systems to the manufacturing processes [15].

As mentioned above, *modularity* and *customization* are two of the key characteristics of RMS. They were given significant attention in our work. Arai et al. [16] proposed the “Plug & Produce” (P&P) concept, which enables new devices to be introduced in a manufacturing system easily and quickly. Chen [17] extended the idea of modular manufacturing devices by developing a modular robot arm that can be configured on demand. Maeda et al. [18] presented a reconfigurable multi-robot system, where new robots can be introduced to an already running manufacturing process. Neuman et al. [19] present and highlight the importance of a high-level, skill-based control architecture of a robotic cell equipped with P&P modules.

Hardware modularity made possible by the P&P concept signifies that more standardized communication architectures are needed to enable communication between robots and other workcell components. In the recent years, the open-source Robot Operating System (ROS) has been gaining traction not only among the scientific community but also among various robot manufacturers [20]. Despite its name, ROS is not limited to solve only robot communication and control challenges but also provides tools, libraries and conventions to facilitate robot operation at different levels.

Due to their high dexterity, flexibility and a generally large workspace, collaborative robot arms are a good fit for the RMS paradigm. This type of robots are certified to work in close proximity with human workers and do not require a barrier around them to remain safe for the human [21]. They present opportunities to advance beyond standard methodologies for robot programming, which still primarily relies on utilising a robot teach pendant directly connected to the robot controller (on-line programming) or on simulation (off-line programming). Both of these processes are rather unintuitive, tedious, and require knowledge of the specific robot or simulation system. The lack of robot integration in manufacturing processes that are prone to frequent changes is attributed to these issues [22].

A programming method that aims to close this gap is the Programming by Demonstration (PbD) paradigm [23,24]. Among different approaches to PbD, kinesthetic guidance has been widely adopted on collaborative robots [25]. Kinesthetic guidance is defined as a process where a human operator holds the robot and physically guides it through the desired movement. It is effective especially for programming robot skills [26,27], but also for other tasks such as workcell calibration. It is most effective with modern torque-controlled robots. However, as noted by Villani et al. [28], even modern collaborative robots do not offer intuitive enough programming interfaces that would allow for fast re-purposing of the robot. In this paper, we explain how the PbD paradigm based on kinesthetic teaching can be tightly integrated with reconfigurable hardware and ROS-based software infrastructure to realize a workcell that can be set-up for new production tasks quickly.

We have introduced the basic concepts and ideas of a reconfigurable robot workcell in [29]. Since then, we expanded this initially proposed system with additional passive hardware components and systematically implemented the methodologies to reconfigure passive hardware with robots. We have also streamlined the process of teaching new robot skills and enhanced programming of complex state machines.

3. Reconfigurable hardware and workcell design

When developing a Reconfigurable Manufacturing System (RMS), e.g. a reconfigurable robot workcell, it is necessary to consider the desired physical properties of the overall system, e. g. size, robustness of the structure, the available robot workspace, safety, etc. The available factory space plays a significant role in determining the workcell layout. Furthermore, it is important to ensure that the cell can be integrated into an existing manufacturing process without making too many changes to the said process. When changes do occur, the workcell needs to be adaptable to cope with them quickly. Finally, the reconfigurability should not significantly increase the price of the automated solution.

To comply with these requirements, we followed the concept of passive reconfigurable components. The main reasoning behind this idea is that a robot workcell already contains an active component, namely a robot(s), which can be used to (re)configure the rest of the robot’s workspace. Thus hardware components with passive degrees of freedom can be manipulated by a robot and moved to a configuration suitable for the desired task. They are a more affordable option compared to high-cost active solutions, which are often prohibitively expensive for manufacturing SMEs that want to keep the costs of automation low. Besides the fully automated reconfiguration, the cell should also support manual reconfiguration when full automation is not feasible.

In the following we describe several passive hardware components that we developed to facilitate reconfiguration and adaptation to new production demands.

3.1. Reconfigurable frame

The frame of the workcell is a structure that connects the robot with peripheral modules that provide different functionalities. The main design requirement for the frame is stiffness. Stiffness is important for robotic applications because even small frame deformations can result

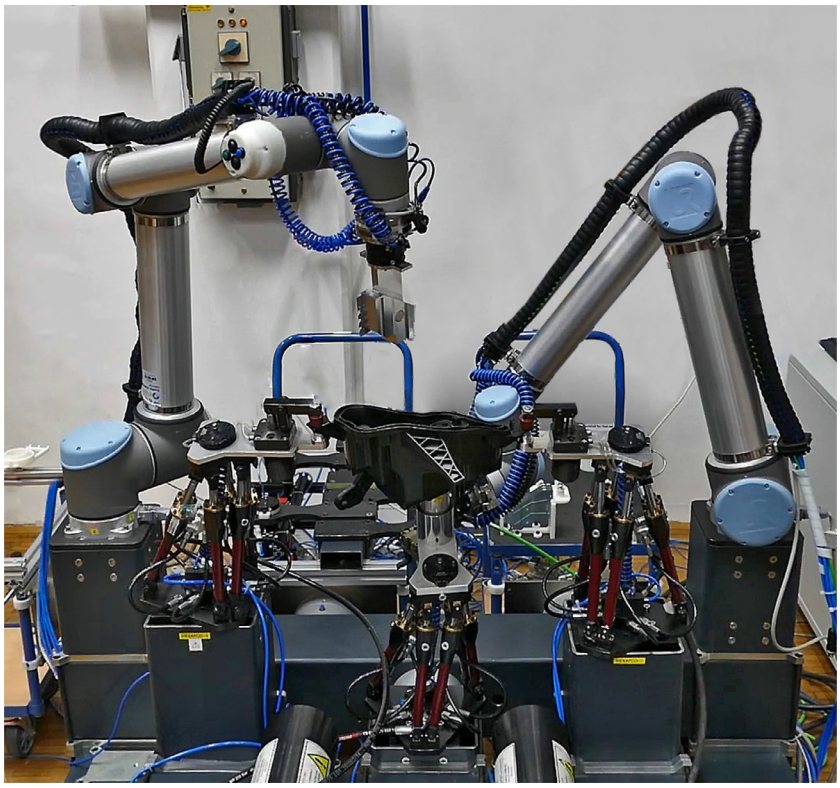


Fig. 1. The reconfigurable robot workcell in a configuration where it assembles an automotive headlight. The cell frame is constructed from steel beams, held together by the BoxJoint system. Peripheral modules are connected to the cell via P&P connectors, e. g. blue trolleys in the back and the flexible fixtures module in the front. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

in large positioning errors, especially in the case of peg-in-hole like operations. On the other hand, when a major change in the production process occurs, we must be able to adapt the workcell's structure to make it compatible with a new product. To comply with these requirements, the frame has to be sufficiently flexible (i. e. adaptable) so that changes can be made quickly (Fig. 1).

To fulfil all these requirements, a workcell frame made of rectangular steel beams was chosen. The beams are connected using an innovative system developed for the aeronautical industry called BoxJoint [30]. The resulting frame structure is very stiff and comparable with welded joints and at the same time simple to assemble and modify by workers. Similar characteristics can be achieved using some other solutions, for example Bosch Rexroth or similar systems. However, off-the-shelf solutions that use extruded aluminium profiles cannot achieve the same level of stiffness and are not as affordable.

3.2. Plug & produce connectivity

Peripheral modules are crucial for the operation of the reconfigurable robot workcell as a manufacturing system. A robot workcell without any peripheral elements cannot be used to perform any real production tasks. Thus every workcell should have the ability to be augmented with peripheral elements (modules) that provide the appropriate functionalities. Some typical modules include fixtures, tool storage, material flow management, and other application-specific equipment. To ensure smooth process flow and short reconfiguration times, we need to have the ability to introduce these modules into the workcell or swap them with others (especially the material flow modules) as quickly and with as little interruption to the manufacturing process as possible.

To deliver the ability to quickly add and remove these modules, we developed special "Plug & Produce" (P&P) connectors (see Fig. 2). These are designed to provide quick mechanical coupling with highly repeatable and stiff positioning of the peripheral modules. In order for these peripheral components to be truly modular, they should be self-sufficient to a certain degree, i. e. they should be equipped with the appropriate computing, actuation and other capabilities. It is therefore not enough to provide only mechanical coupling but also power and connectivity. For this reason, the developed P&P connector also provides electrical power, Ethernet connection for data transfer, and pneumatic lines, which can all be used by the equipment contained within the module. This in turn enables the hardware modules to be completely self-sufficient and ready to provide the desired functionality as soon as they are coupled to the main frame. Moreover, the developed module provides stiff and accurate coupling, which means that when a new peripheral module is attached, (re)calibration of the workcell is not necessary provided the locations of equipment and parts within the newly attached module are known. While there are commercially available solutions that offer similar features to the developed P&P connector, e. g. from Destaco, they are rather expensive for what is needed when connecting external periphery. This is mainly due to higher positioning repeatability and more connectivity options. The simplistic design of the developed P&P connector assures that it can be manufactured at a lower cost. On the other hand, the lower repeatability is compensated in our system by using the novel robot finger design algorithms that are described in Section 3.4.3.

This working principle is not limited to peripheral modules but can be used for the robot's end-effectors as well. In the proposed workcell, all end-effectors utilize the Destaco QC/TP-30 quick tool exchange system. The working principle of the tool exchange system is similar to that of P&P connectors. It provides mechanical coupling of the end-effectors with the robot as well as electricity and pneumatic air. The active part that is mounted on the robot incorporates a ball clutch, which is engaged or disengaged using pneumatic air. This allows the robots to quickly and fully autonomously exchange the tools and other end-effectors and at the same time allows the application of more advanced end-effectors, described in more detail in Section 3.4. Additionally, the tool exchange system can be used to kinematically calibrate the cell and the robots by attaching the male ends of the tool exchange system to a hardware module which position is not known in the world coordinate system. The calibration is then preformed by kinesthetically guiding the robot and attaching its end-effector to the

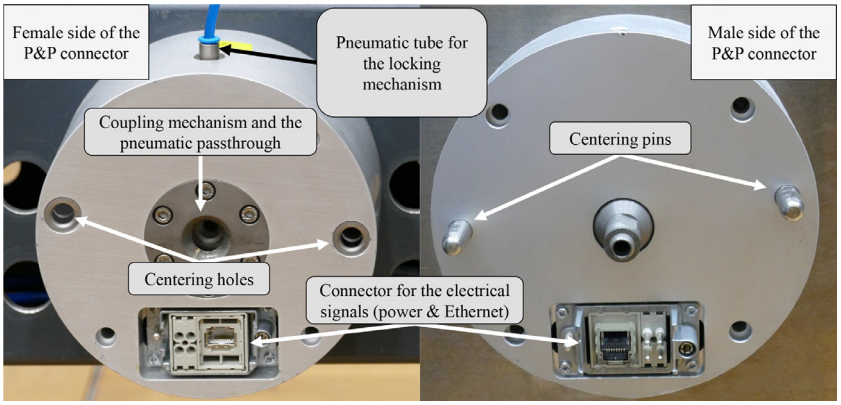


Fig. 2. Female (right) and male (left) end of the developed "Plug & Produce" (P&P) coupling system.

coupling elements placed in the workcell. This way we can calibrate the simulation to match the real cell (described in Section 6.1) or calibrate the base frame of a multi-robot system (documented in [31]).

3.3. Passive reconfigurable hardware components

While the plug "Plug & Produce" type of connectors allow us to introduce new modules into the workcell, thus modifying and enriching its functionality, such modules often need to be introduced manually, i. e. by a human worker. This type of reconfiguration is not autonomous. Standard off-the-shelf solutions for autonomous reconfiguration require active components and high precision measuring equipment, which often leads to a high price tag. To lower the price, we propose to follow a novel concept of passive reconfigurable hardware components. The proposed passive elements do not contain any actuators or sensing equipment. Instead, since every robot workcell contains a robot, the robot's manipulation and sensing capabilities are used to carry out reconfiguration and positional sensing. We evaluated this approach by developing a number of passive hardware components that are described in the following sections.

3.3.1. Passive linear rail

Linear rails provide the means to increase the robot's workspace by moving the robot's base along the rail. The rail units are usually actuated so this process can be done autonomously, but this in turn raises their price. A solution proposed in this work is to use a passive linear unit, along which the robot can move its base using its own actuators. A set of brakes ensure that the base is fixed in place when they are engaged (see Fig. 3a). The robot can reconfigure its base position by latching onto a fixed anchor point with its end-effector and then propelling itself along the linear rail (see Fig. 3b and c). During this operation the brakes have to be disengaged and are re-engaged only when the base of the robot reaches the desired position. In compliance with the reconfigurability paradigm, we utilized the male part of the tool exchange system (described in Section 3.4) as the anchor point. The time it takes for such reconfiguration depends on specifications of the robot used: robots with a lower payload will take more time as accelerations have to be smaller to keep the joints torques within the limits. Generally, it takes a couple of seconds (4–12) for the full reconfiguration so it is mostly beneficial in cases where cycle times are not critical.

We evaluated the developed system by performing a series of experiments with a Franka Emika Panda robot. We chose this robot for this experiment because – unlike the UR10 robots used in other experiments – it is equipped with torque sensors in every joint. This allowed us to measure the strains on the robot's joints during the motion along the linear rail. Our experiments showed that the torques at the joints are within the specified limits for all joints, as shown in Fig. 3d. We also measured the positioning repeatability of the robot's base along the rail using a precision gauge (accurate up to 0.01 mm). In this experiment, the robot moved from one end of the rail to the other where it touched the precision gauge contact point. Our tests have shown that the positioning repeatability of the robot base along the rail is $\pm$ 0.2 mm. This means that by using the developed system, we do not significantly worsen the accuracy at which the robot can perform the assembly operations, which is 0.1 mm according to the specification of the robot. Furthermore, the positioning errors of the base can be compensated in part since in our system, the tasks are programmed relative to the fixed anchor point and we can compute the position of the anchor point using forward kinematics.

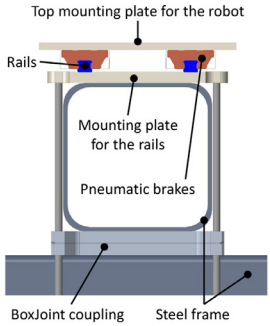
3.3.2. Passive flexible fixtures – hexapods

A manufacturing process that requires a workpiece to be precisely and firmly fixed before certain assembly operations are performed, can be implemented using specially designed fixtures. The design and manufacturing of fixtures can amount to 10% – 20% of the total manufacturing system costs [32]. These costs can be reduced by passively reconfigurable solutions. Our approach is to use an unactuated Gough-Stewart platform as the base for fixture elements [33]. These fixtures can be, upon releasing the brakes, moved by the robot arm, which amounts to changing the position and orientation of the top of the Gough-Stewart platform. When the desired new position is reached, the brakes are engaged. The robot system stores the last known position of the fixture before fully releasing it. Just like in the case of linear rails, the male end of the tool exchange system is placed on the top plate of the Gough-Stewart platform while its female counterpart is attached to the robot's end-effector. Thus the robot can attach itself to the top of the Gough-Stewart platform by engaging the tool exchange system. A more detailed description and evaluation of the proposed passive flexible fixtures is provided in [34].

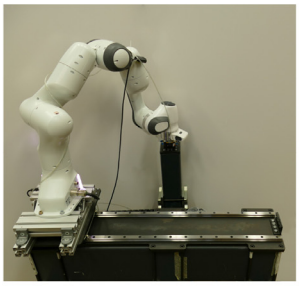
A combination of the proposed unactuated Gough-Stewart platforms (in our experiments we usually used three of them) can be used to generate a fixture system for workpieces of different shapes. A robot can automatically reconfigure such a system to different configurations suitable for fixing different workpieces. An example of automatic reconfiguration of a fixture system for two different automotive light housings is shown in Fig. 4.

3.3.3. Passive rotary table

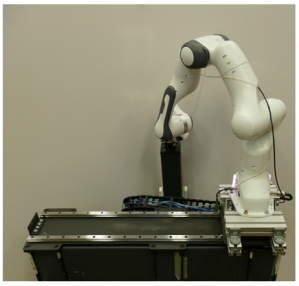
Sometimes it is necessary for a robot arm to perform assembly and other operations from different sides of the workpiece. We designed a passive rotary table to address the issue when the robots workspace does not allow the robot's end-effector to reach the workpiece from all relevant sides. The workpiece is placed on the table that can be rotated by a robot to move the workpiece to the desired orientation (an example assembly task is shown in Fig. 5b). The actuation and sensing is again accomplished by using the robot's actuators and sensors. A good



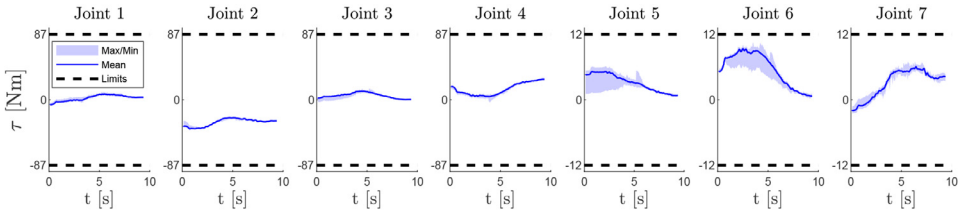
(a) A cross section of the passive linear rail.



(b) Robot on the left side of the passive linear rail.



(c) Robot moved to the right side of the passive linear rail.



(d) The arising toques during reconfiguration (solid blue lines) and the torque limits (dashed).

Fig. 3. The developed passive linear rail allows for re-positioning of the robot's base to extend its workspace. The robot is mounted on a platform that can slide along rails and contains pneumatically activated brakes, which disengage before the robot is to be moved. For autonomous re-positioning, the robot attaches itself to the frame with the tool exchange system and then propels its base to the desired new position (Fig. 3b and c). Fig. 3d shows that the arising joint torques are within the limits for all joints.

positioning repeatability is achieved with pistons pushing plungers into the holes of the table top to fix it (a cross section is depicted in Fig. 5a). Consequently, the table can be fixed only at a finite number of discrete orientations. The pistons are retracted when the workpiece needs to be re-oriented. Thus pistons and plungers provide the same functionality as pneumatic brakes in case of linear rails and reconfigurable fixtures. The proposed solution is cheaper to manufacture than standard rotary tables with active actuation and sensors.

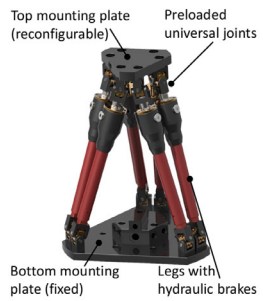
3.4. Reconfigurable robot tools

Traditionally, a tool mounted on the robot's end-effector is rarely exchanged. Once the tasks of the robot is defined, the robot is equipped with the appropriate tool for the specific task. Naturally, this approach is not adequate for a reconfigurable robot workcell. A step towards solving this issue is to make use of the "Plug & Produce" connectivity in the form of a tool exchange system attached to the end-effector, as described in Section 3.2. This can greatly expand the array of tasks that can be performed by a robot without the need to manually make changes to its end-effector. However, with an eye on the affordability of the overall solution and execution times, we wanted to minimize the number of needed grippers and other specialized robot equipment. In this section, we present two reconfigurable tools that were developed in order to reduce the number of costly pieces of equipment by providing a degree of reconfigurability on the tool itself.

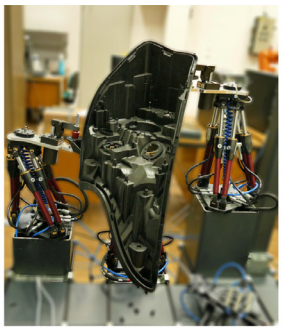
3.4.1. Reconfigurable screwdriver

Automated screwdrivers are well established in manufacturing as they provide an efficient way to fasten screws to predefined values, i. e. angles or torques. The commercially available industrial screwdriver solutions are highly tailored to specific tasks and cannot be reused for another application (e. g. different screws) without modifications. Such modifications are trivial to do for a human but very challenging for a robot. The lack of flexibility combined with the high price tag motivated us to develop a screwdriver solution that provides means for the robot to autonomously exchange the screw bits.

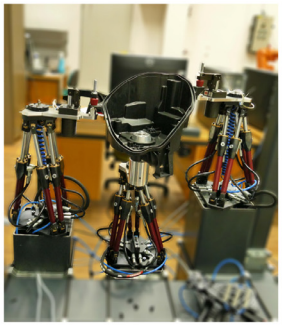
We enhanced an off-the-shelf industrial screwdriver spindle (Deprag Minimat-EC) with a custom designed power transmission that allows for a quick exchange of screw bits. The developed solution for the screw bit exchange consists of two main components, the power transmission shaft and the screw bit plug, which attaches to the shaft. A secure connection between the shaft and the screw bit plug is achieved using a pneumatically activated coupling system, while the angular lock is achieved using a simplified spline joint. On the other end, the screw bit



(a) Computer render of the passive flexible fixture.

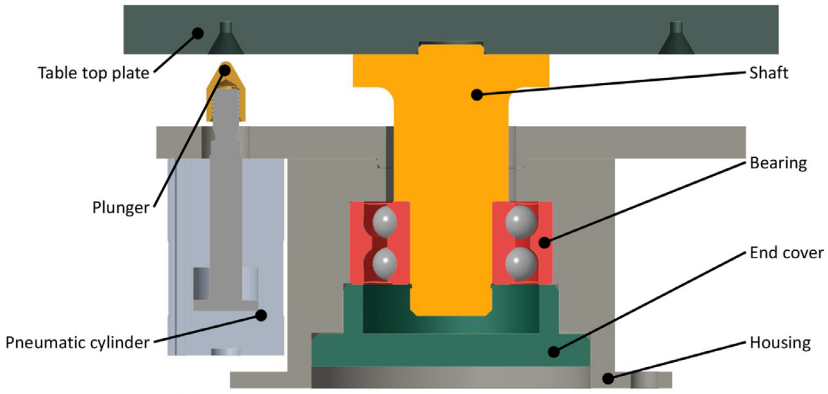


(b) Fixtures configured for the X87 headlight housing.

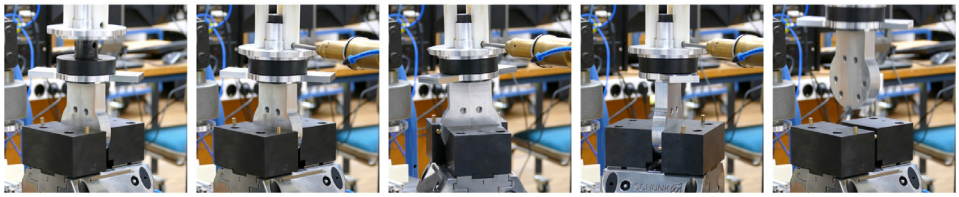


(c) Fixtures configured for the X07 headlight housing.

Fig. 4. A passive fixture is designed as a Gough-Stewart platform, containing hydraulic brakes in each leg so that the legs can hold the top platform firmly in the desired position. It is possible to mount various fixturing elements (e. g. centering cones or clamps) on the top platform. To automate the re-positioning of these fixtures, we also installed the male end of the tool exchange system, thus allowing the robot to move the top end of the fixtures when the brakes are released.



(a) A cross section of the passive rotary table.



(b) A sequence (left to right) showcasing the use of the passive rotary table to fasten screws on three different sides of a circular workpiece.

Fig. 5. Passive rotary table that enables re-orientation of a workpiece in order to ensure that a robot can perform assembly operations from all sides of the workipece. The cross-section of the rotary table is depicted in Fig. 5a, whereas Fig. 5b shows a sequence of orientations of the table that ensure that a workpiece is placed in such a way that the robot can fasten screws from 3 different sides of the workpiece (the applied screwdriver is described in Section 3.4.1).

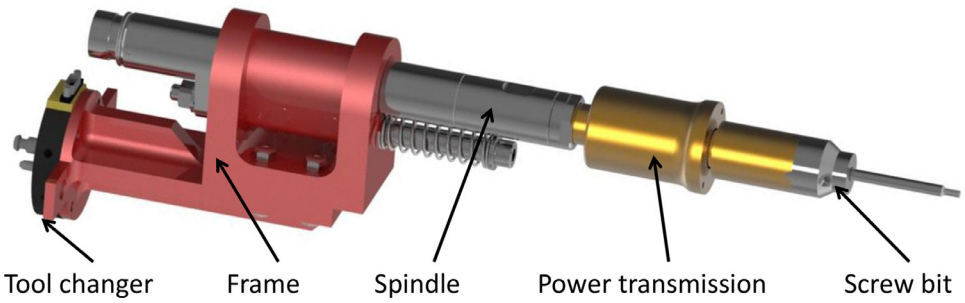


Fig. 6. A computer render of the reconfigurable robotic screwdriver. It is based on a commercially available screwdriver spindle, which we enhanced with a specially developed power transmission that allows for a quick exchange of screw bits. A custom made frame around the spindle provides a mounting point for the tool exchange system.

plugs can be fitted with any screw bit that follows the DIN 3126-E6.3 shank standard. Other shank shapes could be utilized with only minor modifications. By mounting this power transmission to the screwdriver, the robot was able to exchange the screw bits. The complete screwdriver with the screw bit exchange system is depicted in Fig. 6.

3.4.2. Reconfigurable gripper fingers

Equipping robots with tool exchange systems is a step that adds much needed flexibility in terms of expanding the array of operations that they can perform without manual intervention. While providing a large array of end-effectors increases the flexibility, such an approach also increases the overall costs, which works against our goal to provide an affordable system. This is especially problematic for grasping because automated robot assembly requires that workpieces are grasped at precise, known poses. Otherwise assembly operations that require high precision cannot be executed. Precise grasping is difficult to achieve with current state of the art dexterous hands, therefore they are still rarely used in industrial applications [9]. Instead, specialized grippers are often constructed for each different workpiece. To address this problem, we developed a finger exchange system that can reduce the number of robotic grippers needed in the cell for grasping workpieces of different shapes [35].

The developed finger exchange system is shown in Fig. 7. It consists of pneumatic couplers mounted on the robotic gripper and finger tips that can be exchanged on demand. By providing such a system it becomes possible to use one gripper to grasp objects of different shapes as long as their sizes are similar.

3.4.3. Position error compensation with smart finger design

To achieve precise robot part manipulation, the fingertips of the robotic gripper should be carefully designed. Current design approaches are time consuming, lack intuitiveness, and are inflexible in terms of reusability. To enable fast construction of new fingers, we developed a new approach to design, manufacture, evaluate, and re-design gripper fingers [36]. Our framework consists of automated design methods, evaluation through simulation, and iterative optimization of the initially designed fingers.

We extended the previously developed automatic finger design frameworks to not only deal with two-finger servo grippers but also with three-finger pneumatic grippers. In contrast to designing fingers for a servo gripper, in the case of pneumatic grippers several constraints (e. g. the opening and closing width of the gripper, exact part width, closing force) must be considered in the design process. We utilized two primary methods for this purpose. The first one uses the imprint of the part's geometry to define the relevant finger shape features, while the second method uses a set of simple geometrical descriptors describing the features of the object.

The initial design, acquired by either method, is then evaluated in a dynamic simulation from which we estimate the performance of the design. If the performance meets the specified success criteria, the designed fingers are manufactured with a 3D printer and introduced into the workcell. On the other hand, if the performance does not satisfy the

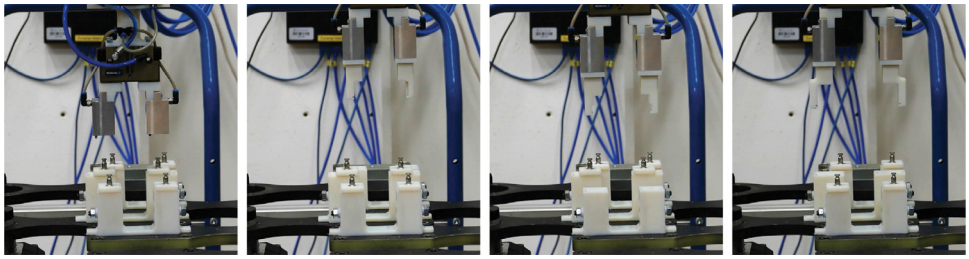


Fig. 7. A finger exchange system allows the robot to grasp of objects of different shapes without needing to exchange the gripper. The pictures depict the finger exchange system with no fingers (left most picture) and with 3 different fingers attached.

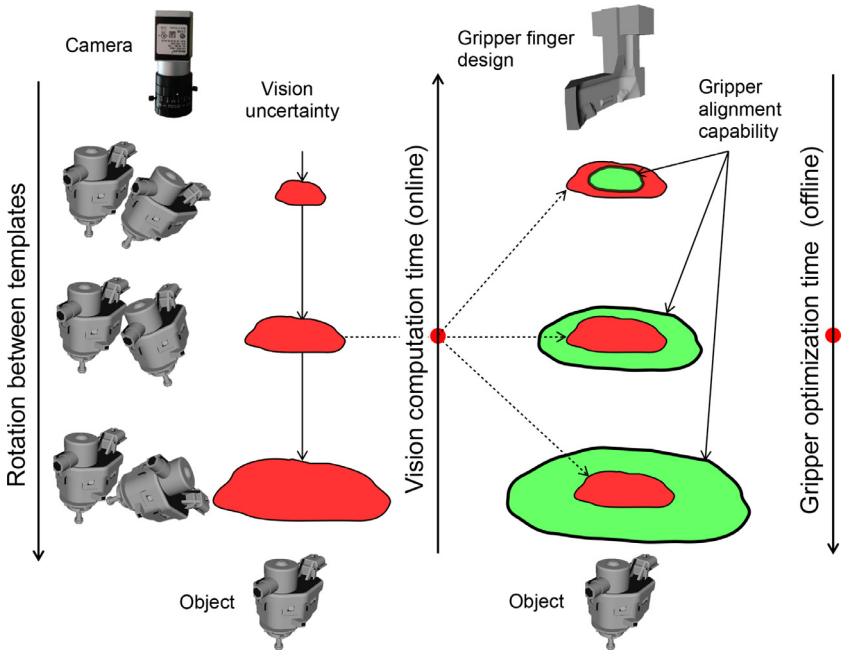


Fig. 8. Graphical representation of the combined vision and finger design optimization framework.

task requirements, the finger design is further optimized until the requirements are met.

In the majority of industrial manipulation tasks, the uncertainties about the workpiece poses are related to the vision-based pose estimation. For this reason, we developed a novel framework for analysing the relation between the pose estimation uncertainty and the grasping uncertainty for manipulation and assembly purposes. The framework (see Fig. 8) consists of two methods:

- the first method deals with optimising the vision system's parameters to reduce pose estimation uncertainty based on a set of parameters related to vision information processing,
- the second method addresses off-line optimization of finger shapes to make the fingers capable of compensating for the vision system uncertainty.

The combined output of the two methods gives an optimal solution for the desired task. The evaluation of the proposed methodology shows that both the optimization of vision parameters and the optimization of finger design can contribute to improving the reliability of grasping [37]. However, the results can be further improved when the two methods are used in combination. This is important because more accurate pose estimation usually require longer processing times. Thus we can stop vision processing once sufficient precision for the required grasping accuracy has been achieved. On the other hand, it is not necessary to further optimize finger design once the designed fingers can deal with the achieved accuracy of pose estimation.

4. Software architecture of a reconfigurable robot workcell

Besides providing physical connections between the peripheral modules as described in Section 3, it is also important to ensure connectivity in the context of data flow. Each peripheral module should be connected to the same network in order to broadcast its data and receive information and instructions about what action to perform at any given time. In this section we present a ROS-based software architecture that – complementary to modular hardware design – ensures software modularity. An overview of the developed software system architecture is shown in Fig. 9. Its components and how they complement the reconfigurable hardware are described in more detail throughout this section.

4.1. The workcell ROS backbone

Simply ensuring the data flow between various modules within the workcell is not enough to adhere to the requirement of software modularity. In addition, we need to ensure that the data is structured in such a way that all modules within the system can parse them. For example, data containing the measurements of the force-torque sensor mounted on the robot's end-effector should be readable by all software modules within the system without the need to code specific parsers on all the receivers.

In this respect, the Robot Operating System (ROS) represents a suitable framework for developing various software components that need to share data over the shared network. The various tools and

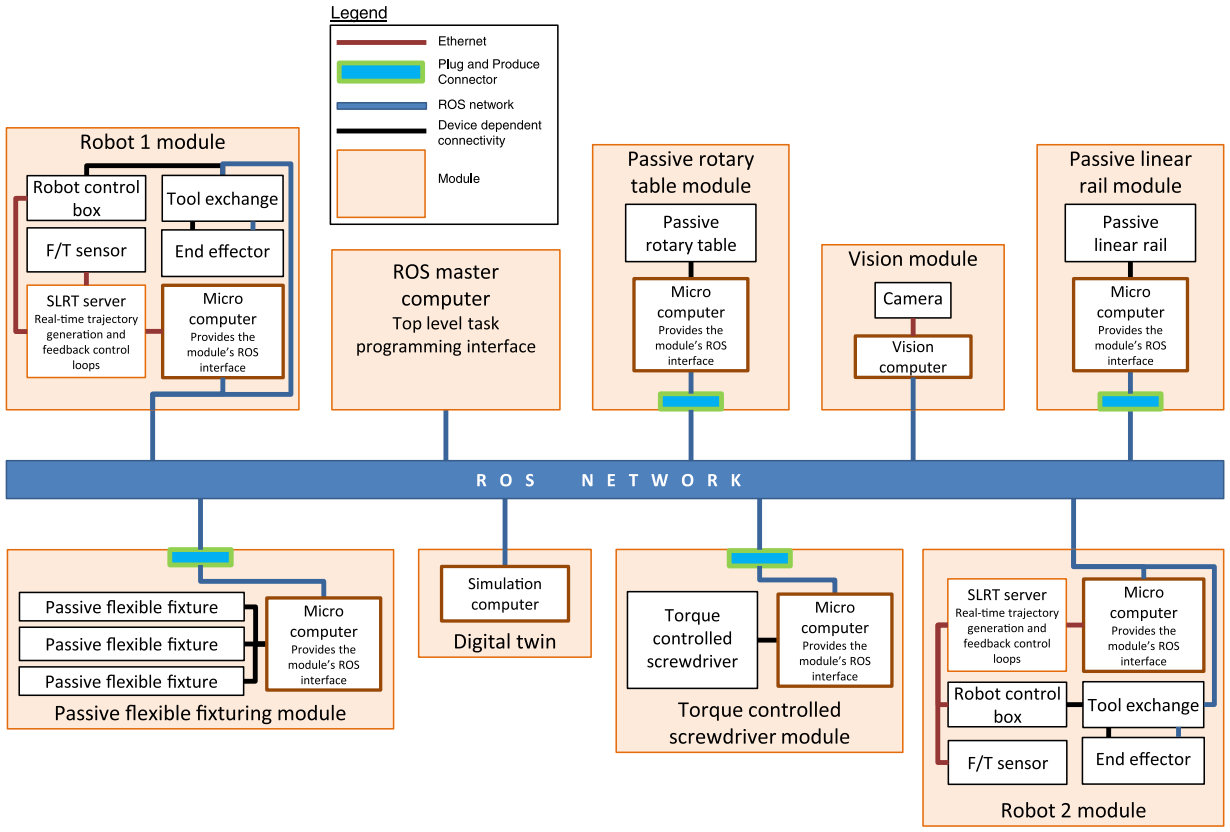


Fig. 9. Software architecture of the reconfigurable robot workcell with various software and hardware modules. Every module within the cell is designed in such a way that it connects directly to the ROS network. This ensures that every module within the workcell can provide and receive data using the same communication format. To further expand this connectivity paradigm, all hardware components are equipped with Ethernet interfaces that provide ROS connectivity to them. This includes the tool exchange system, a smart servo gripper, robotic screwdriver, etc.

features that are available within ROS enabled us to achieve the pursued software reconfigurability of the cell. In our case, software reconfigurability means that it is possible to expand the cell's functionalities without disrupting the current software architecture. New software components can be developed without the need to reprogram any of the existing ROS nodes. This also eases the development and integration of new hardware components with their own ROS nodes, as described in Section 4.2.

4.2. Workcell modules and ROS

Adding or removing modules and therefore reconfiguring the workcell should not require the designer to dedicate a lot of time and attention to the software connectivity. Our driving paradigm was that each module within the workcell should be able to connect to the ROS network. Thus all modules should be equipped with sufficient computational hardware to run ROS nodes, thereby exposing each module's data and functionalities to the workcell's ROS network. This way the modules can be controlled by the top-level task scheduling software as soon as they are connected to the workcell. Some modules connected to the cell can require more than just network connectivity in order to function properly, e. g. pneumatic air or electric power. In such cases, the modules must be connected using the developed "Plug & Produce" connector, which is mounted either on the workcell's frame or on the robot's end-effector (as described in Section 3). These design characteristics coincide with the RMS design guidelines that advise for software and hardware modularity.

To illustrate how a module connects to and is managed by the ROS network, we describe the *torque controlled screwdriver* module in more detail. This module is based on the Deprag Minimat-EC screwdriver connected to the Deprag AST-11 *Sequence controller* [38]. To trigger a screwing sequence, a digital signal has to be sent to the *Sequence controller*'s digital interface. As the *Sequence controller* cannot be programmed to host ROS *nodes*, it was necessary to connect it with something that can. For this purpose, we selected a Linux-based single-board development computer to host ROS nodes that provide a ROS interface to the *Sequence controller* of the screwdriver. The digital outputs of the single-board computer are connected to the *Sequence controller*, thus enabling a bidirectional data exchange between the two. The developed ROS *nodes* on the single-board computer thus provide a ROS interface to the *Sequence controller*. A call to the ROS *Action Server* running on the single-board computer sends the pre-defined digital signals to the *Sequence controller*, which consequently triggers one of the screwing sequences (fasten to torque, fasten to angle, etc.). The module is connected to the workcell with the "P&P" connector as it requires more than just Ethernet connection. This module is depicted in Fig. 9 as *Torque controlled screwdriver module*.

4.3. Low level real-time robot control

A robot manipulator should integrate into the cell without breaking the RMS principles. This means that it should be able to integrate with the rest of the hardware components seamlessly. Most of the industrial robots are equipped with a control box that, apart from ensuring real-time control of the robot, also provides a task programming interface. However, these control boxes do not support running ROS nodes so a special communication layer to connect the robot with the rest of the ROS network is needed. We therefore developed an abstraction layer that supports switching between different types of robots. This layer provides a number of trajectory and feedback control strategies independently of the selected robot and enables the programming of new strategies via a suitable control interface. This design decision is

compliant with our overall design concept, i. e. a robot is just another module within the workcell and should therefore be easily replaceable. To support real-time control, the proposed abstraction layer was developed as a real-time server that on one side communicates with the selected robot at the highest frequency allowed by the robot's control box, and on the other side with a ROS node (called robot ROS driver) responsible to communicate with the ROS network.

The real-time server is based on Matlab Simulink Real-Time Target (denoted as SLRT Server in Fig. 9) and accepts high-level commands, e. g. action sequences, from the robot ROS driver and applies the implemented control strategies and commands to execute the required robot motion. The server also broadcasts information about the state of the robot and hardware connected directly to the robot control box (e. g. 6-D force-torque sensor) to the robot ROS driver. The robot ROS driver is a ROS node running on a microcomputer within the robot module and connects to the ROS network via the Ethernet interface (see Robot 1 and 2 module in Fig. 9).

The development of a custom real-time server that controls the robot motions independently of the robot's controller has also allowed us to implement custom trajectory generation strategies and advanced control algorithms that are usually not found in controllers provided by robot manufacturers.

5. Technologies for fast programming of assembly operations

In order for the robot workcell to successfully carry out an assembly operation, it is necessary to compile a top-level program that schedules each of the tasks performed by the cell according to the product assembly specification. This step does not only take up a significant portion of the setup time but usually also demands proficiency and know-how in programming and robotics. Accelerating this process is therefore essential for enabling fast setup and short reconfiguration times. On the other hand, increasing the ease of use and intuitiveness of the programming process is also required in order to allow shop-floor workers to partake in the setup process. In this section we present technologies that were developed and implemented in order to address these challenges. More specifically, we address the problem of programming robot movements and skills when interfacing with the reconfigurable hardware as well as high-level task programming.

5.1. Acquisition of robot assembly skills by human demonstration

The definition of robot motions to carry out a complete assembly process can be difficult and time consuming for non-expert users. Programming by Demonstration (PbD) provides a methodology to define these motions in a natural way rather than by coding complex programs in a robot programming language. In the proposed cell, PbD is based on kinesthetic guidance, which enables the user to move the robot through its workspace by physically guiding it along the desired path.

To further improve on the intuitiveness and speed of programming by demonstration, we equipped the robot with a button interface. For this purpose, the cover of one of the robot's joints was replaced by a custom-made, 3D-printed cover that houses several programmable buttons and switches (shown in Fig. 10). This way we gain the possibility to program the buttons and switches for various purposes that facilitate the PbD process. In our setup, we programmed the buttons and switches as follows:

- Switch 1 – Gravity compensation mode toggle.
- Switch 2 – Lock/Unlock toggle for the tool exchange system.
- Button A (blue) – Mark the current robot configuration for saving.
- Button B (green) – Open/Close toggle for the air flow to the tool exchange system.

5.2. Robot assembly skill database

Before programming a sequence of robot actions that leads to the complete assembly, the robot operator should equip the robot with the necessary assembly skills. To make the skills available to all components of the workcell, the *mongodb_store* ROS package has been integrated into the workcell's control system. This way we enabled all ROS nodes in the network to access a MongoDB database [29]. In our setup, the MongoDB database runs on the ROS master computer. As described in Section 5.1, skills are acquired by means of human demonstration. Thus acquiring a new skill means demonstrating the desired robot motion and saving it as a named new entry into the MongoDB database. For point-to-point movements, fixed robot configurations are saved, whereas complex trajectories are saved as parameters of dynamic movement primitives (DMPs), which is a flexible representation for complex roobt movements. See [39,40] for the discussion of the advantages of DMPs. It then becomes possible to define a high-level assembly sequence that reads these named entries (robot configurations or DMPs) from the database and moves the robot accordingly. The poses and trajectories are saved in the database as ROS messages corresponding to each type of movement. Having these skills saved as named entries enables quick reconfiguration in terms of changing skills. It is sufficient to overwrite the entry in the database with a modified skill to update the assembly sequence without changing the high-level assembly sequence program.

5.3. State machine assembler

The next step after setting up the robot workcell, both in terms of hardware modules and robot skills, is to prepare the assembly sequence as a computer program. While there are many libraries and frameworks that facilitate coding of complex state machines, they all essentially require high degree of programming proficiency, which is against our guiding principle to make workcell programming easier. Visual programming languages represent an alternative to hand writing code. A widely used tool for visual programming within the ROS community is *FlexBE* [41]. While it does provide a comprehensive GUI that requires the user to form "connections" between various "blocks" to define a ROS-based state machine, it lacks in terms of reconfigurability. All the components that make up these blocks have to be pre-loaded on the development computer.

To tackle these issues in a manner that preserves the principle of software reconfigurability, we developed State Machine Assembler (SMACHA) framework that can compile executable programs based on top-level scripts and low-level templates. The top-level scripts have to be written in YAML data-serialization language, while templating is done using Jinja2 templating language. The details of how this framework works are presented in [42]. Here we highlight how the functionality of loading templates to the ROS parameter server is used to support hardware reconfigurability as well.

The SMACHA's code generation engine shown in Fig. 11 is able to read templates that are used to generate executable code not only from files on the development computer but from the ROS parameter server. This gives us the possibility to develop the workcell's modules that upload their templates to the ROS parameter server upon connecting to the ROS network. At a first glance, this feature seems redundant as most of the modules within our system already provide a ROS interface either via *Action Servers* or *Services* (detailed in Section 4.2). However, the main advantage of using templates is that they can do more then just make a request to the *Action Servers* or *Services*. They can also perform data manipulation, conditional statements, etc.

6. Reconfiguration

There are several aspects of reconfiguration in the proposed workcell. We start by discussing reconfiguration to completely new

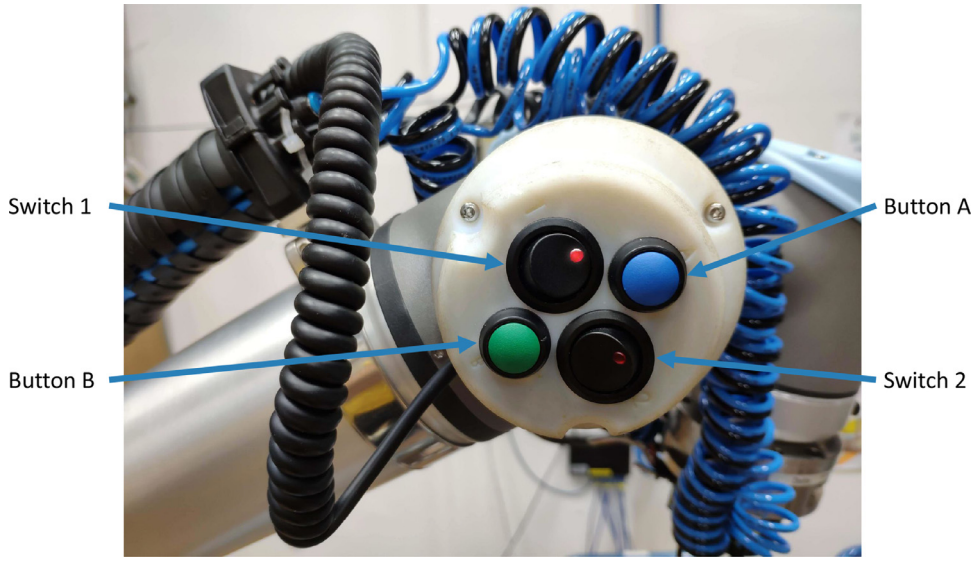


Fig. 10. Custom-made, 3-D printed button interface. The replacement cover houses two buttons and two switches. The latter have a LED showing their status (in the picture switch 1 is turned on). These buttons and switches are programmable and their functions can differ depending on the skill acquisition process.

production tasks, which usually requires some manual steps, e. g. restructuring the workcell frame, adding and removing some peripheral hardware components from the workcell, etc. Once a new component has been introduced (trolleys, tools, etc.), the workcell must be calibrated, i. e. the pose of the newly introduced component with respect to the robot's base coordinate system need to be acquired. For this purpose, one end of the tool exchange system is attached to the new component and the other end to the robot's end-effector. The robot is kinesthetically guided to the new hardware component and attached to the tool exchange system part. The tool exchange system is engaged to force the robot into the final desired pose (by connecting the two parts of the tool changer) and the pose of the robot's end-effector is computed from the robot joint angles to compute the pose of a new hardware component.

As described in Section 3.3, passive reconfigurable components have no actuators or sensors installed in them, therefore they cannot move on their own. However, these parts can be automatically reconfigured by a robot as they contain passive degrees of freedom. We use a tool exchange system, kinesthetic guidance, and point-to-point movements to program reconfiguration movements. In all cases (passive linear rail, fixtures, rotary table), we gather data by attaching one part of the tool exchange system to the robot's end-effector. In the case of a linear rail, the other part is attached to the fixed holding location in the workcell (see Fig. 3). The robot is first kinesthetically guided towards the holding locations and attached to the tool changer. This initial robot pose is recorded. The linear rail brakes are then released and the robot base is kinesthetically guided towards the desired pose. Once the desired pose has been reached, the brakes are engaged and the final robot pose is stored. Out of these data, the automatic reconfiguration program can be generated: 1. move the end-effector towards the initial holding location using a simple point-to-point-movement, followed by a linear approach movement with a fixed orientation, 2. connect the two parts of the tool changer, 3. release the brakes of the linear rails, 4. move the base along the linear path from the initial to the final pose, 5. engage the brakes, 6. release the connection between the two parts of the tool changer, 7. withdraw the robot's end-effector away from the tool changer part. This programming process is fully supported by a button interface shown in Fig. 10.

The procedure is similar for the passive fixtures (hexapods) and the rotary table.

6.1. Reconfiguration in simulation

The manual determination of a suitable workcell layout, which includes locations of all physical components of the cell, is difficult to perform and takes a considerable amount of time. Here we describe

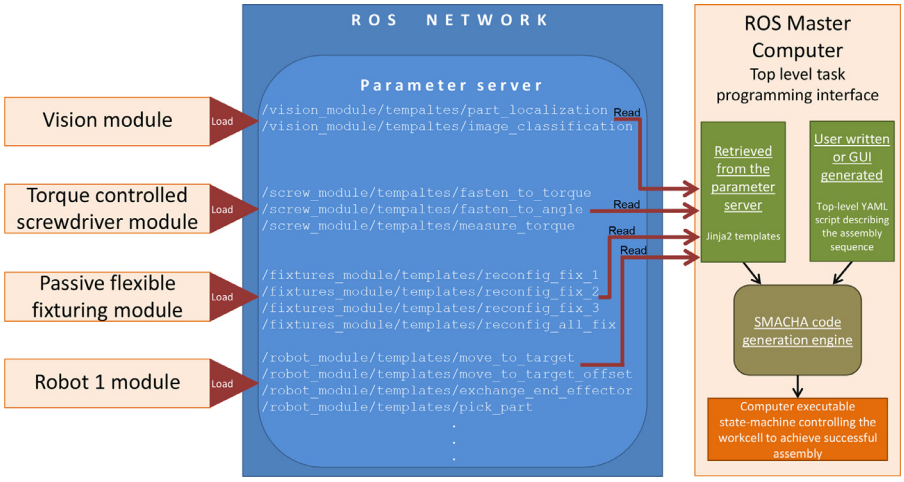


Fig. 11. SMACHA's modular software design supplements the modular hardware design of the robot workcell. Each module within the cell can upload SMACHA templates to the parameter server. This scheme shows an example how the *Vision module*, the *Torque controlled screwdriver module*, the *Passive flexible fixture module* and the *Robot 1 module* upload their templates to the parameter server. The robot programmer writes a top-level YAML script describing the assembly sequence as a state machine using the templates available on the parameter server. The SMACHA generator then combines these inputs and generates the appropriate computer executable code.

how the developed system supports the (re)configuration process by making use of a digital twin.

The digital twin we developed supports the modelling of the cell from a hardware and software point of view [43]. It connects to the ROS software infrastructure and can access the SMACHA templates and scripts. This allows the user to program, simulate and run the assembly sequence with the relevant robot motions in simulation [44]. It also provides a graphical user interface showing the 3D representation of the cell. The workcell designers can use the GUI to determine the layout of the cell's hardware components for the given production task. This includes the placement of the robot(s), end-effector tools, periphery elements and the arrangement of the passive reconfigurable modules, etc. This is especially useful for industrial users because it is not necessary to stop production when constructing a cell for the next production task.

However, the main advantage of being able to simulate robot motion is the ability to autonomously evaluate different layouts. After the initial layout has been determined in such a way that the robot(s) can perform the desired task, the workcell designer can proceed to fine tune the positions of various hardware elements to achieve a more optimal solution. During this process, the chosen components are moved to new locations (in the vicinity of the initial ones) and the assembly process is simulated. The assembly is simulated for every new location of the chosen components and the relevant metrics are monitored (e. g. cycle times, energy consumption, etc.). The respective measures are shown in the GUI.

Some example measures are shown in Fig. 12. In Fig. 12a we evaluated how the position of the robot's base affects the cycle time of the assembly. This evaluation measure has showed us that the initially chosen position for the robot base was not optimal. We therefore moved the robot's base to the optimal location suggested by the cycle time metrics. In another experiment, we evaluated the location of the workpiece that the robot had to manipulate. This time we monitored the total energy consumption to perform the required robot motions. The results are shown in Fig. 12b. It is evident that the closer the workpiece is to the base of the robot, the lower the total energy consumption is. While this could be expected, the workcell designer can extract additional information from these measurements. We can observe that the energy consumption rises faster along the *Y* axis of the workpiece position. In case the closest position is not viable to place the workpiece, it is better to move it along the *X* axis than the *Y* axis.

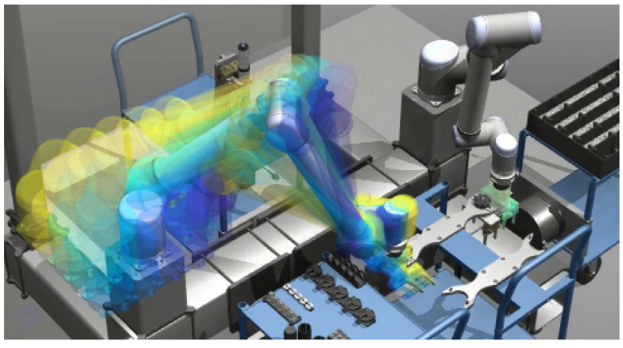
7. Integration for fast setup of robot assembly processes

Following the RMS paradigm and its characteristics, we introduced several innovative technologies to make our system – apart from reconfigurable – also *modular* and *customisable*. We achieved that by designing the overall system to be constructed using modular hardware, complemented by a modular software architecture. On the hardware side, this comprises (Section 3): 1. several module exchange systems including a novel P&P connector that guarantees stable mechanical coupling and also passthrough of data and power, 2. passively reconfigurable elements, e.g. fixtures based on Gough-Stewart platform with hydraulic brakes, 3. reconfigurable tools, e.g. a robotic screwdriver with exchangeable screw bits, 4. design of gripper fingers based on 3-D printing, etc.

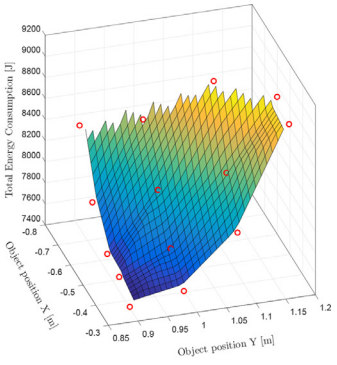
To provide the characteristics of an RMS, the cell is also equipped with several advanced software tools and frameworks (Section 5): 1. kinesthetic guidance supported by a button interface to ease the acquisition of new assembly skills and data for calibration, 2. a new scripting and templating framework for high-level task programming, and 3. digital twin component integrated with the real workcell to support programming and reconfiguration. The developed workcell control system (Section 4) is based on ROS, which fully supports the modular design of the cell, including synchronization and communication between all active components.

In the following we briefly describe the workflow that leads to the implementation of a new production task using the proposed system. We also outline how to reconfigure the cell when changes to the production task emerge. A diagram showing the pipeline for deploying a new production task to the reconfigurable robot workcell is shown in Fig. 13.

When a new production task needs to be implemented, the first step is to gather information and specifications about the desired product. This step is typically performed in cooperation with the client. The process of setting up the workcell is an iterative process. After the initial analysis has been performed, the workcell users start preparing the solution from the available software and hardware components (marked blue and pink in Fig. 13, respectively). On the hardware side, the cell structure is assembled/reconfigured using the reconfigurable frame parts, the appropriate peripheral modules (including passive reconfigurable components and external sensors), and the required robot tools. These steps can first be done in the digital twin environment to produce an initial draft of the cell layout. On the software side, the assembly sequence is prepared by making use of the available robot skills, by acquiring new skills by human demonstration, and by using the high-level task programming framework. The developed system also supports the integration of vision-based methods for object recognition and pose estimation, which are needed if the location and identity of workpieces are not fixed [37]. Similarly, visual quality control approaches can be integrated using the graphical programming methods described in our previous work [45].



(a) Cycle time length depending on different positions of the robot base.



(b) Energy consumption when manipulating an object at different positions.

Fig. 12. The digital twin can be used to support the design of optimal workcell layout. Within simulation we can evaluate how different placements of hardware components affect the relevant performance indicators. Fig. 12a shows how different positions of the robot's base affect the cycle time of the assembly (darker shades – lower, lighter shades – higher). In Fig. 12b we can see the effects of workpiece location on the total energy consumption when the robot picks up the workpiece. Here the object was picked up from 14 different locations on a plane (marked with red circles). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

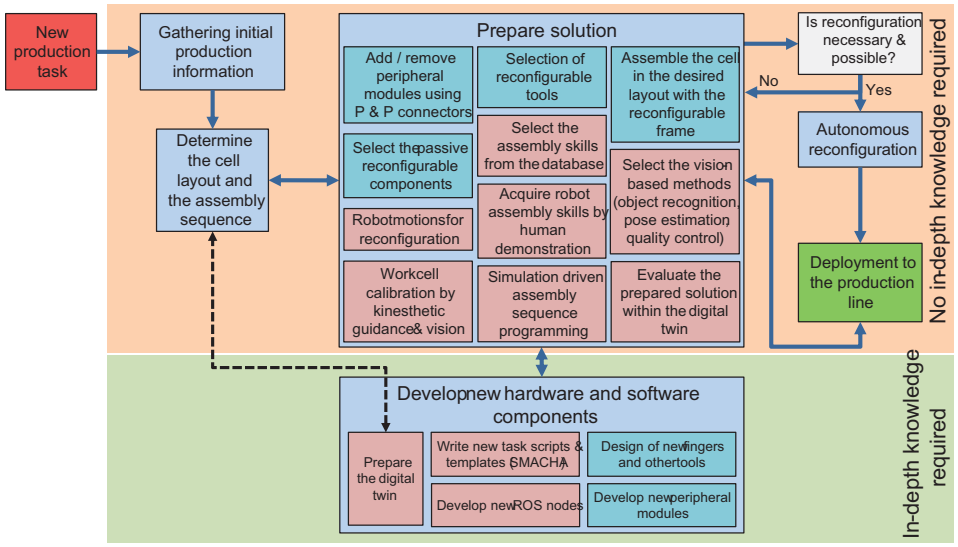


Fig. 13. The workflow for deploying new production tasks in the reconfigurable robot workcell. The developed technologies assure that most of the integration tasks can be performed without in-depth knowledge. However, if the production task require new capabilities or modules, users with more in-depth knowledge can develop new software and hardware components without disrupting the already defined workcell architecture.

The required robot skills can be initialized and the workcell calibrated by kinesthetic guidance, which is supported by a button interface to ease the programming tasks. Similarly to the cell hardware layout, the initial assembly sequence can be programmed in the digital twin as well. If the solution needs to be reconfigured automatically to switch the production process, e. g. by manipulating passive peripheral components, the necessary robot movements should also be programmed. Finally, the solution is deployed to the real production environment. In most cases, some adjustments still have to be made after the first deployment to optimize the performance.

The processes described above can be carried out without in-depth knowledge of the system. However, if it is not possible to prepare a solution from the already available components, the personnel with in-depth knowledge can develop new hardware or software modules. On the hardware side, this means either developing new peripheral modules or designing new robot tools and fingers. By providing the P&P connectivity, the new hardware components can be attached to the cell with ease. On the software side, new SMACHA scripts and templates or ROS nodes can be developed.

While the digital twin is not strictly necessary to prepare a new solution, its availability can accelerate the set up process. However, the preparation of the digital twin is usually not possible without in-depth knowledge about the desired product and the operation of the workcell.

8. Evaluation

To evaluate the developed workcell and the implemented methodologies, we performed a series of experiments in relevant environments, realising five industrial production processes from different manufacturing areas. Reference Key Performance Indicators (KPIs) were acquired and the overall performance of the cell was evaluated. Throughout the experiments, some of the key equipment stayed the same, but other parts of the workcell were reconfigured according to the requirements of each task. Some application-specific peripheral modules were either added or removed.

All experiments were carried out in close cooperation with the manufacturing companies that provided the use cases. The most important KPIs were defined as: 1. quality of assembly, 2. reconfiguration time, 3. time to implement an automated assembly solution, and 4. cycle time to assemble the products. In this section we present the evaluation of these crucial indicators.

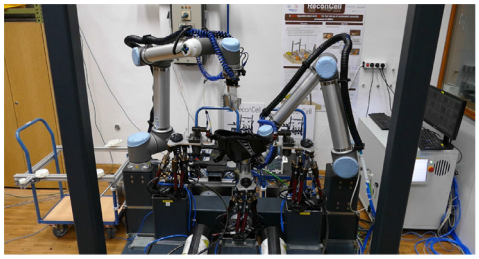
8.1. Setup times

We define setup times as the time it takes to setup a new production process in the developed workcell. It is composed of several steps, including the exchange of information with the customer, the production and evaluation of the hardware designs, and the development of the required software (see Fig. 13). Evaluating the efficiency of workcell setup and its reconfigurability in a manner that provides quantitative results is difficult due to the lack of standardized benchmarks. We therefore approached this evaluation by implementing various use cases from different manufacturing areas. The goal was to show that not only different manufacturing processes can be automated within the developed workcell, but that it is also possible to switch from one production process to another by exploiting the workcell's modular design, its reconfigurability, and without making major changes to the core software and hardware structure of the cell.

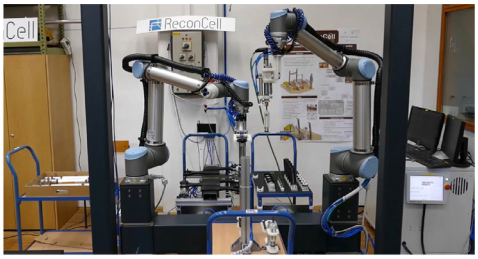
The following five different production processes were implemented: 1) assembly of different automotive light housings, 2) assembly of a customized linear actuator for smart furniture, 3) assembly of a glass mounting gripper, 4) assembly of a family of airport runway signalling lights, and 5) assembly of various versions of an electronic device by inserting multiple PCB modules into their housing. The variants of the workcell configured for each of these experiments are shown in Fig. 14 and in videos added as supplementary material to the paper [46–50].

We estimated the duration of implementation of the latter use cases with the available resources (the first two use cases were developed in parallel with the development of workcell software infrastructure, thus their implementation took longer than necessary). Because a full implementation consists of integrating both hardware and software, we estimated the time it took for a full implementation by considering the data from a version-control software system. In Table 1 we compare the dates from the first and the last commit of the code for the top-level state machine. Even though such data stem from the work done on the software, the hardware work was done in parallel. The implementation of use cases showed that the setup duration is most affected by hardware work, as it was always necessary to develop new tools and peripheral modules.

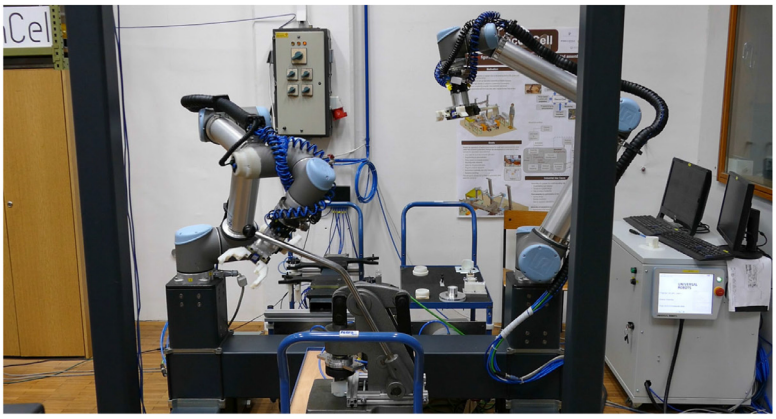
From these data and experience gathered during the implementation, we can conclude that focused implementation of different use cases can be achieved in two weeks to one month. If all the needed hardware is available from the beginning, it is usually possible to design the cell and implement the assembly task in 2 – 4 days.



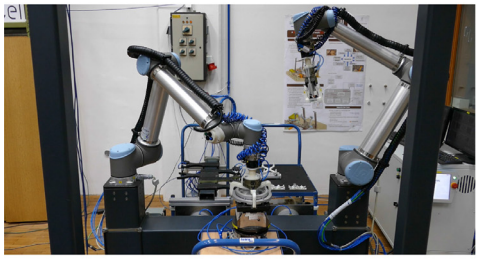
(a) Assembly of automotive light housings.



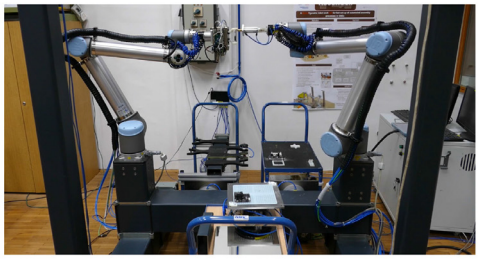
(b) Assembly of a customized linear actuator for smart furniture.



(c) Assembly of a glass mounting gripper.



(d) Assembly of a family of airport runway signalling lights.



(e) Assembly of electronic devices.

Fig. 14. The five industrial use cases implemented in the proposed reconfigurable workcell. The series of images demonstrates that that some of the peripheral components stayed the same while some of the modules had to be exchanged. Additionally, different tools were made available on the tool rack for the robots to use.

Table 1
Estimated duration of use case implementations. The second use case includes Christmas holidays, thus the duration of its implementation is somewhat skewed.

Use case	Beginning	End	Total duration
Glass mounting gripper assembly	29-10-2018	06-12-2018	40 days
Runway lights assembly	06-12-2018	05-02-2019	61 days
Electronic device assembly	11-02-2019	26-02-2019	15 days

8.2. Reconfiguration times & cycle times

The proposed workcell supports different forms of reconfiguration. First we studied switching the production from one product variant to another. For example, in case of changing the production from one automotive light housing to another (see also Section 8.3), reconfiguration is performed by a robot physically moving the passively reconfigurable fixtures into a suitable configuration to fix the current light housing for assembly. In case of assembly of various versions of an electronic device, reconfiguration consists of generating a new assembly program based on identity of devices entering the cell and exchanging the gripper fingers to enable grasping the workpieces. We were able to perform these and other reconfigurations fully automatically in a matter of minutes.

Another form of reconfiguration involves switching the workcell from one production case to another. We tested several combinations, including switching from the assembly of automotive light housings to the assembly of the customized linear actuator for smart furniture and switching from the assembly of runway lights to the assembly of electronic devices. This type of reconfiguration cannot be performed fully automatically and therefore involves some manual work, e.g. to rebuild the workcell frame and bring new tools to the workcell. Nevertheless, provided all hardware and software components are available, this type of reconfiguration can be performed in a matter of tens of minutes.

The focus of our work is to achieve a high degree of reconfigurability of the system as a whole and provide automation to production processes, which are to a large degree still done manually. Nonetheless, we did measure also the cycle times achieved in our use case implementations. They are presented in Table 2. Some of the achieved cycle times cannot be compared to their real-life counterparts because we implemented the assembly of prototypes, which are not yet assembled in a regular production line. In those cases where cycle times

Table 2
Cycle times of each use case implemented in the reconfigurable workcell. Only some of the implemented production processes have a real-life counterpart production running in a factory. Therefore, only those ones have the corresponding cycle times presented in brackets.

Description of the assembly	Achieved cycle times
Automotive light housing (X07)	2'25" [1'05"]
Automotive light housing (X82)	1'53" [41"]
Glass mounting gripper	4'51"
Customized linear actuator	6'53" [1']
Runway light (square casings)	1'51"
Runway light (round casings)	1'22"
Electronic device assembly	4'47"

of the automated production could be measured, they are longer than manual production but in two of the three use cases comparable and actually acceptable for the company interested in automating its production. It has been noted also by other researchers that – compared to more tailor-made solutions – RMS usually achieve lower throughput [8,13]. Shorter cycle times could be achieved by further optimising the hardware and the workcell layout for each use case, but this was not the focus of our work.

For the third use case where we could compare the cycle times, i. e. assembly of customized linear actuator, the automated solution takes too much time. In this case the conclusion was that the parts need to be redesigned to enable automated assembly in real production. This is a well known problem in automation; short cycle times can rarely be achieved if the requirements of automation are not considered during the product design process [51].

8.3. Robustness and quality of automotive light housings assembly

As explained above, our experiments stem from different areas of manufacturing. For a more thorough analysis, we selected the assembly of light housings from the automotive industry. The automotive industry has very strict and well-defined requirements in terms of quality of the production process, which makes it easier to select the industrially relevant key performance indicators (KPIs).

The implemented manufacturing process revolves around the assembly of automotive light housings for headlights. Two different headlight models (X82 and X07) were selected for this experiment, both shown in Fig. 15. The assembly process involves the insertion of various components into the headlight housing. The headlights mainly differ in shape. However, also the parts to be inserted are different (see Fig. 15b).

In its current implementation on the factory floor, the assembly process is in part performed manually, with a final operation performed by a specially designed assembly machine. Each light housing model requires a different assembly machine. Typically 2 workers are needed to service these machines in a required cycle time and the production takes place in four shifts (24/7), i. e. 8 workers each day. While these machines provide an assembly method with short cycle times, they are not cost effective as they have to be designed and produced for each headlight model. Additionally, these assembly machines have to be available for 5 years after the end of regular production, as spare parts must be produced on demand, occupying significant storage space.

The aim of the experiment was to evaluate the proposed reconfigurable workcell as a fully automated substitute for the numerous assembly machines and manual work needed in the current version of the assembly line. This means that the workcell has to be able to assemble different types of light housings without human intervention. As noted before, these headlight models differ not only in their shape, but also in their assembly sequence. The challenge of the different shapes was addressed by implementing the passive flexible fixtures (hexapods) described in Section 3.3.2 and depicted in Fig. 4b and c. Furthermore, the difference in the assembly sequence was addressed by providing an array of different robot tools, mounted on the tool rack module of the workcell. The robots were thus able to autonomously reconfigure the fixtures and equip themselves with the tools needed for each step in the assembly process.

8.3.1. Methodology

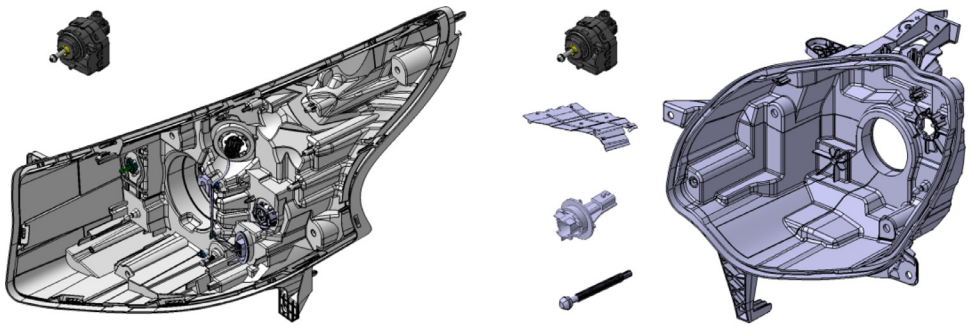
To evaluate the suitability of the workcell for this task, we conducted the *Run@Rate* analysis, which consists of equipment confirmation and process confirmation. The goal of *Run@Rate* is to identify potential quality and/or productivity problems and put countermeasures in place to prevent these issues affecting further development prior to the start of production. Measuring System Analysis (MSA) and Production Part Approval Process (PPAP) are a standard part of the *Run@Rate* analysis, with the guidelines provided by the Automotive Industry Action Group [52]. While MSA is used for the confirmation of equipment, the confirmation of the process is done using the PPAP.

During the PPAP test, the parts assembled in the workcell were monitored and the relevant KPIs extracted. Some KPIs required the measurement of physical values, e. g. heat shield and screw height, which was performed by using specialized measuring equipment. Other KPIs, e. g. the condition of LWR drive and the bulb holder, required visual inspection of the possible damage to the headlight housing due to faulty insertion. These KPIs are listed in Table 3.

We studied the robustness of the manufacturing process implemented in our workcell with respect to the production upper (*USL*) and lower (*LSL*) specification limits. To quantify the results of the test, the following two capability process indices with continuous values were used as KPIs:

- C_p, which estimates what the process is capable of producing if the process mean were to be centred between the specification limits of the process:

$$C_p = \frac{USL - LSL}{6\sigma}, \tag{1}$$



(a) Automotive headlight model X82 (b) Automotive headlight model X07

Fig. 15. Two different automotive light housings that were assembled in the proposed workcell and the respective parts that need to be inserted in order to complete the assembly. The left side figure shows the headlight model X82 where only the LWR drive has to be inserted. The right hand figure shows the headlight model X07 with the parts to be inserted (from top to bottom): the LWR drive, a heat shield, a bulb holder and a screw.

Table 3
Key performance indicators for the assembly of automotive light housings.

Part	KPI description
Heat shield	Height between the heat shield after insertion [mm]
Screw height	Height of the screw after successful fastening [mm]
LWR drive	Possible material damage due to faulty insertion [OK / NOK]
Bulb holder	Possible material damage due to faulty insertion [OK / NOK]

Table 4
Results from the statistical analysis of the production part approval process (PPAP).

	Heat shield	Screw	LWR drive	Bulb holder
LSL [mm]	10.50	21.80	/	/
USL [mm]	11.50	22.80	/	/
Process target [mm]	11.00	22.30	/	/
Success rate [%]	100.00	100.00	100.00	100.00
Process mean - μ [mm]	10.97	22.36	/	/
Standard dev. - σ [mm]	0.11	0.10	/	/
C_p	1.49	1.63	/	/
C_{pk}	1.40	1.40	/	/

- C_{pk}, which estimates what the process is capable of producing considering that the process mean may not be centred between the specification limits:

$$C_{pk} = \min\left[\frac{USL - \mu}{3\sigma}, \frac{\mu - LSL}{3\sigma}\right]. \quad (2)$$

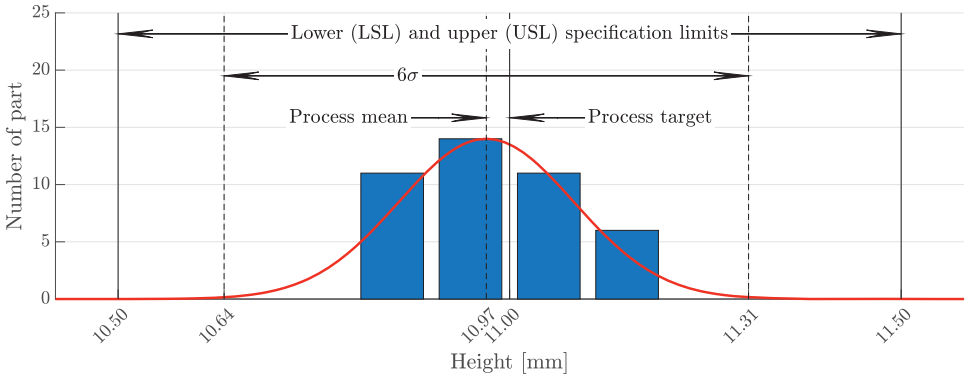
In the above equations, σ represents the variability of the process (standard deviation) and μ represents the estimated process mean. A high C_{pk} indicates that the process is adequate and has a small spread in relation to the tolerance width. If C_{pk} is equal to C_p, then the process is set to produce exactly in the middle of the tolerance range.

8.3.2. Results

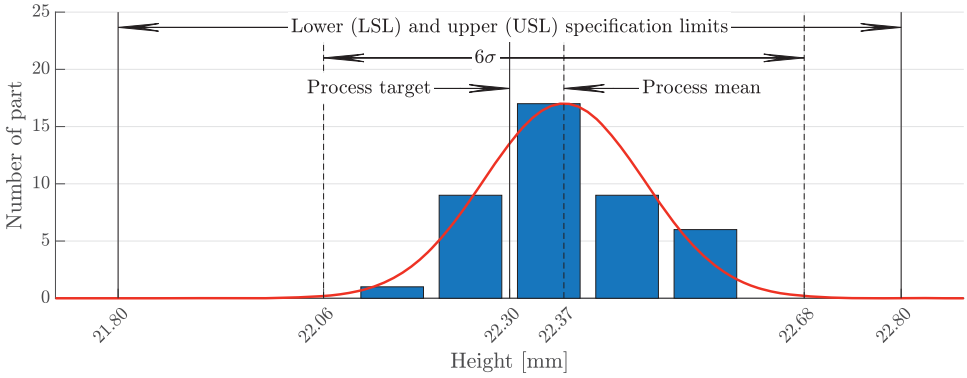
The confirmation of the equipment using MSA was performed by assessing and confirming the equipment used for the assembly. For example, the flexible fixturing system was confirmed by concluding that it can ensure the fixturing of good parts, whereas faulty parts cannot be put onto the fixturing system. For the various grippers, their suitable performance was confirmed by the correct grasping of good parts, whereas faulty parts could not be grasped correctly.

After the equipment was confirmed, we carried on to the Production Part Approval Process (PPAP). In this experiment, 40 pairs of headlights had to be assembled continuously one after the other. The measurements and visual inspection of the KPIs was done by a representative from the cooperating company. From the acquired data, it was then possible to calculate C_p and C_{pk} indices for the screw and heat shield height KPI. The LWR drive and bulb holder KPIs are expressed in a binary form (OK or NOK), hence it does not make sense to perform statistical analysis beyond the success rate for these two KPIs. The measurements taken during the evaluation are presented in the top part of Table 4. The results of benchmarking are provided in the bottom part of Table 4, while Fig. 16 expresses these results graphically.

Typically, the value for C_{pk} should be at least 1.33 [53]. For the conformation of the assembly process, C_{pk} must be above 1.33. In our case we achieved $C_{pk} = 1.40$ for both of the measured, non-binary KPIs. This means that both MSA and PPAP were confirmed and the *Run@Rate* test was successful. Thus we have proven that the quality of assembly is appropriate to deploy the developed solution on a factory floor.



(a) Graphical representation of the heat shield height statistical analysis.



(b) Graphical representation of the screw height statistical analysis.

Fig. 16. The graphs depict the statistical analysis of the screw and heat shield height measurements. The blue bars represent the histogram of the measured values, while the red line represents an estimated Gaussian distribution for the said measurements. From the graphs it is evident that both KPIs fall within the specified lower and upper limit. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

9. Conclusion

In this paper we present a Reconfigurable Manufacturing System (RMS) developed as a reconfigurable robot workcell. Our work focused on the synergies between the hardware and software modules to ensure a high degree of reconfigurability, short setup times, and quick reactions to production changes. Throughout the development of the cell, we paid significant attention to improve affordability and ease of use of the overall system. We introduced the concept of passive reconfigurable hardware components to achieve affordable and autonomous reconfiguration. Further achievements presented in this work include the integration of programming by demonstration practices into the workcell setup streamline, the standardization of hardware interfaces with the "Plug & Produce" connectors, hardware modularity supported by the ROS-based software architecture, and the application of digital twin in order to adapt and optimize the layout of the cell.

The proposed technologies and the overall system were evaluated by implementing several real industrial production tasks. The successful implementation of these experiments demonstrates the versatility of the proposed system and its components. In one of the experiments, i. e. assembly of automotive light housings, we were able to confirm the industrial grade quality of the implemented production process.

In a large system like the one described in this paper, there are of course many aspects that can be further improved. One important aspect is the overall safety of the proposed robot workcell. As described in the paper, the system makes use of collaborative robots that are certified to be safe to operate in proximity to human workers. However, the developed passive reconfigurable components, which use the robot arm as the sensing and actuation mechanism, do not necessarily share the safety mechanisms of a collaborative robot. For example, the legs of the passive flexible fixtures have a gap between them that can change significantly and unpredictably when it is moved by a robot. This has the potential of harming a person's finger and represents a safety hazard for the person unaware of this risk. A possible solution for this specific issue would be a protective membrane that would cover the legs of the fixture without hindering its flexibility. However, when designing such solutions it is important to maintain a high degree of reconfigurability of the overall system.

Thus one important task for future research is to provide procedures to ensure safety and define certification processes for reconfigurable workcells. Since reconfigurable workcells are not static but change frequently, it is necessary to define appropriate certification procedures that will enable fast certification of new variants of the workcell, including the certification with respect to safety issues.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work has received funding from the EU's Horizon 2020 innovation action ReconCell (grant agreement no. 680431).

Supplementary material

Supplementary material associated with this article can be found, in the online version, at10.1016/j.rcim.2020.101979.

References

[1] Y. Koren, U. Heisel, F. Jovane, T. Moriwaki, G. Pritschow, A.G. Ulsoy, H.M. Van Brüssel, Reconfigurable manufacturing systems, CIRP Ann. 48 (2) (1999) 527–540.
[2] Z.M. Bi, S.Y.T. Lang, M. Verner, P. Orban, Development of reconfigurable machines, Int. J. Adv. Manuf.Technol. 39 (11) (2008) 1227–1251.
[3] M.G. Mehrabi, A.G. Ulsoy, Y. Koren, Reconfigurable manufacturing systems: key to future manufacturing, J. Intell. Manuf. 11 (4) (2000) 403–419.
[4] H.A. ElMaraghy, Flexible and reconfigurable manufacturing systems paradigms, Int. J. Flexible Manuf. Syst. 17 (4) (2005) 261–276.
[5] Z.M. Bi, S.Y.T. Lang, W. Shen, L. Wang, Reconfigurable manufacturing systems: the state of the art, Int. J. Prod. Res. 46 (4) (2008) 967–992.
[6] Y. Koren, M. Shpitalni, Design of reconfigurable manufacturing systems, J. Manuf. Syst. 29 (4) (2010) 130–141.
[7] H. ElMaraghy, G. Schuh, W. ElMaraghy, F. Piller, P. Schönsleben, M. Tseng, A. Bernard, Product variety management, CIRP Ann. 62 (2) (2013) 629–652.
[8] G. Zhang, R. Liu, L. Gong, Q. Huang, An analytical comparison on cost and performance among DMS, AMS, FMS and RMS, in: A.I. Dashchenko (Ed.), Reconfigurable Manufacturing Systems and Transformable Factories, Springer, Berlin, Heidelberg, 2006, pp. 659–673.
[9] N. Krüger, A. Ude, H.G. Petersen, B. Nemec, L.-P. Ellekilde, T.R. Savarimuthu, J.A. Rytz, K. Fischer, A.G. Buch, D. Kraft, W. Mustafa, E.E. Aksoy, J. Papon, A. Kramberger, F. Wörgötter, Technologies for the fast set-up of automated assembly processes, Künstliche Intelligenz 28 (4) (2014) 305–313.
[10] M. Stevenson, L.C. Hendry, B.G. Kingsman, A review of production planning and control: the applicability of key concepts to the make-to-order industry, Int. J. Prod. Res. 43 (5) (2005) 869–898.
[11] M.J. Land, G.J. Gaalman, Production planning and control in SMEs: time for change, Prod. Plann. Control 20 (7) (2009) 548–558.
[12] T.D. Brunoe, A.-L. Andersen, K. Nielsen, Reconfigurable manufacturing systems in small and medium enterprises, in: J. Bellemare, S. Carrier, K. Nielsen, F.T. Piller (Eds.), Managing Complexity, Springer, 2017, pp. 205–213.
[13] S. Makris, P. Tsarouchi, A.-S. Matthaiakis, A. Athanasatos, X. Chatzigeorgiou, M. Stefos, K. Giavridis, S. Aivaliotis, Dual arm robot in cooperation with humans for flexible assembly, CIRP Ann. 66 (1) (2017) 13–16.
[14] N. Kousi, C. Gkournelos, S. Aivaliotis, C. Giannoulis, G. Michalos, S. Makris, Digital twin for adaptation of robots behavior in flexible robotic assembly lines, Procedia Manuf. 28 (2019) 121–126.
[15] T. Dietz, U. Schneider, M. Barho, S. Oberer-Treitz, M. Drust, R. Hollmann, M. Hägele, Programming system for efficient use of industrial robots for deburring in SME environments, ROBOTIK 2012; 7th German Conference on Robotics, VDE, 2012, pp. 1–6.
[16] T. Arai, Y. Aiyama, Y. Maeda, M. Sugi, J. Ota, Agile assembly system by "plug and produce", CIRP Ann. 49 (1) (2000) 1–4.
[17] I.-M. Chen, Rapid response manufacturing through a rapidly reconfigurable robotic workcell, Robot. Comput. Integr. Manuf. 17 (3) (2001) 199–213.
[18] Y. Maeda, H. Kikuchi, H. Izawa, H. Ogawa, M. Sugi, T. Arai, "Plug & Produce" functions for an easily reconfigurable robotic assembly cell, Assem. Autom. 27 (3) (2007) 253–260.
[19] M. Naumann, K. Wegener, R.D. Schraft, Control architecture for robot cells to enable plug'n'produce, IEEE International Conference on Robotics and Automation (ICRA), Rome, Italy, (2007), pp. 287–292.
[20] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A.Y. Ng, ROS: An open-source robot operating system, ICRA Workshop on Open Source Software, Kobe, Japan, (2009).
[21] P. Akella, M. Peshkin, E. Colgate, W. Wannasuphoprasit, N. Nagesh, J. Wells, S. Holland, T. Pearson, B. Peacock, Cobots for the automobile assembly line, IEEE International Conference on Robotics and Automation (ICRA), Detroit, Michigan, (1999), pp. 728–733.
[22] G.F. Rossano, C. Martinez, M. Hedelind, S. Murphy, T.A. Fuhlbrigge, Easy robot programming concepts: an industrial perspective, IEEE International Conference on Automation Science and Engineering (CASE), (2013), pp. 1119–1126.
[23] R. Dillmann, Teaching and learning of robot tasks via observation of human performance, Rob. Auton. Syst. 47 (2–3) (2004) 109–116.
[24] B.D. Argall, S. Chernova, M. Veloso, B. Browning, A survey of robot learning from demonstration, Rob. Auton. Syst. 57 (5) (2009) 469–483.
[25] M. Hersch, F. Guenter, S. Calinon, A. Billard, Dynamical system modulation for robot learning via kinesthetic demonstrations, IEEE Trans. Rob. 24 (6) (2008) 1463–1467.
[26] M.R. Pedersen, L. Nalpantidis, R.S. Andersen, C. Schou, S. Bøgh, V. Krüger, O. Madsen, Robot skills for manufacturing: from concept to industrial deployment, Rob. Comput. Integr. Manuf. 37 (2016) 282–291.
[27] V. Krueger, F. Rovida, B. Grossmann, R. Petrick, M. Crosby, A. Charzoule, G. Martin Garcia, S. Behnke, C. Toscano, G. Veiga, Testing the vertical and cyber-physical integration of cognitive robots in manufacturing, Rob. Comput. Integr. Manuf. 57 (2019) 213–229.
[28] V. Villani, F. Pini, F. Leali, C. Secchi, Survey on human-robot collaboration in industrial settings: safety, intuitive interfaces and applications, Mechatronics 55 (2018) 248–266.
[29] T. Gašpar, B. Ridge, R. Bevec, M. Bem, I. Kovač, A. Ude, Z. Gosar, Rapid hardware and software reconfiguration in a robotic workcell, 18th International Conference on Advanced Robotics (ICAR), IEEE, 2017, pp. 229–236.
[30] A. Millar, H. Kihlman, Reconfigurable Flexible Tooling for Aerospace Wing Assembly, (SAE Technical Paper 2009-01-3243), (2009), https://doi.org/10.4271/2009-01-3243.
[31] T. Gašpar, R. Bevec, B. Ridge, A. Ude, Base frame calibration of a reconfigurable multi-robot system with kinesthetic guidance, in: N.A. Aspragathos, P.N. Koustoumpardis, V.C. Moulianitis (Eds.), Advances in Service and Industrial Robotics, vol. 67, Springer, 2019, pp. 651–659.
[32] Z.M. Bi, W.J. Zhang, Flexible fixture design and automation: review, issues and future directions, Int. J. Prod. Res. 39 (13) (2001) 2867–2894.

[33] M. Jonsson, G. Ossbahr, Aspects of reconfigurable and flexible fixtures, Prod. Eng. 4 (4) (2010) 333–339.
[34] M. Bem, M. Deniša, T. Gašpar, J. Jereb, R. Bevec, I. Kovač, A. Ude, Reconfigurable fixture evaluation for use in automotive light assembly, 18th International Conference on Advanced Robotics (ICAR), Hong Kong, China, (2017), pp. 61–67.
[35] A. Kramberger, A. Wolniakowski, M.H. Rasmussen, M. Munih, A. Ude, C.S. Schlette, Automatic fingertip exchange system for robotic grasping in flexible production processes, IEEE International Conference on Automation Science and Engineering (CASE), Vancouver, Canada, (2019), pp. 1664–1669.
[36] A. Wolniakowski, A. Gams, L. Kiforenko, A. Kramberger, D. Chrysostomou, O. Madsen, K. Miatliuk, H.G. Petersen, F. Hagelskjær, A.G. Buch, A. Ude, N. Krüger, Compensating pose uncertainties through appropriate gripper finger cutouts, Acta Mech. Autom. 12 (1) (2018) 78–83.
[37] F. Hagelskjær, A. Kramberger, A. Wolniakowski, T.R. Savarimuthu, N. Krüger, Combined optimization of gripper finger design and pose estimation processes for advanced industrial assembly, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macao, China, (2019), pp. 2022–2029.
[38] DEPRAG Screwdriving Technology, (https://deprag.com/en/screwdriving-technology/). Accessed: 2020-01-31.
[39] A.J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, S. Schaal, Dynamical movement primitives: learning attractor models for motor behaviors, Neural Comput. 25 (2) (2013) 328–373.
[40] A. Ude, B. Nemec, T. Petrič, J. Morimoto, Orientation in Cartesian space dynamic movement primitives, IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, (2014), pp. 2997–3004.
[41] P. Schillinger, S. Kohlbrecher, O. von Stryk, Human-robot collaborative high-level control with an application to rescue robotics, IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, (2016), pp. 2796–2802.
[42] B. Ridge, T. Gašpar, A. Ude, Rapid state machine assembly for modular robot control using meta-scripting, templating and code generation, IEEE-RAS 17th International Conference on Humanoid Robots (Humanoids), Birmingham, UK, (2017), pp. 661–668.
[43] M. Schluse, M. Priggemeyer, L. Atorf, J. Rossmann, Experimentable digital twins – streamlining simulation-based systems engineering for industry 4.0, IEEE Trans. Ind. Inf. 14 (4) (2018) 1722–1731.
[44] M. Priggemeyer, D. Losch, J. Roßmann, Interactive calibration and visual programming of reconfigurable robotic workcells, IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Auckland, New Zealand, (2018), pp. 1396–1401.
[45] S. Reich, F. Teich, M. Tamosiunaite, F. Wörgötter, T. Ivanovska, Data-driven approach for general visual quality control in a robotic workcell, 3rd International Conference on Computer Graphics and Digital Image Processing (CGDIP), Rome, Italy, (2019).
[46] Assembly of automotive light housings, (https://youtu.be/OvexJBsAm7g). Accessed: 2020-03-16.
[47] Assembly of a customised linear actuator for smart furniture, (https://youtu.be/9mlv51g0nWs). Accessed: 2020-03-16.
[48] Assembly of a glass mounting gripper, (https://youtu.be/bQUvl2YIINg). Accessed: 2020-03-16.
[49] Assembly of airport runway signalling lights, (https://youtu.be/Xunf83tkn8k). Accessed: 2020-03-16.
[50] Assembly of an electronic device, (https://youtu.be/55QFegH9vCU).Accessed: 2020-03-16.
[51] D. Sanders, Y. Chai Tan, I. Rogers, G.E. Tewkesbury, An expert system for automatic design for assembly, Assem. Autom. 29 (4) (2009) 378–388.
[52] Production Part Approval Process – PPAP, (https://www.aiag.org/quality/automotive-core-tools/ppap). Accessed: 2020-01-31.
[53] D.C. Montgomery, Introduction to Statistical Quality Control, John Wiley & Sons, 2007.