Methoden

Mohamed El-Shamouty, Kilian Kleeberger, Arik Lämmle, and Marco Huber*

Simulation-driven machine learning for robotics and automation

Simulations-basiertes Maschinelles Lernen in der Robotik und Automatisierungstechnik

https://doi.org/10.1515/teme-2019-0072
Received May 15, 2019; accepted August 6, 2019

Abstract: Mass personalization—a megatrend in industrial manufacturing and production—requires fast adaptations of robotics and automation solutions to continually decreasing lot sizes. In this paper, the challenges of applying robot-based automation in a highly individualized production are highlighted. To face these challenges, a framework is proposed that combines latest machine learning (ML) techniques, like deep learning, with high-end physics simulation environments. ML is used for programming and parameterizing machines for a given production task with minimal human intervention. If the simulation environment realistically captures physical properties like forces or elasticity of the real world, it provides a high-quality data source for ML. In doing so, new tasks are mastered in simulation faster than in real-time, while at the same time existing tasks are executed. The functionality of the simulation-driven ML framework is demonstrated on an industrial use case.

Keywords: Mass personalization, machine learning, simulation, robotics, automation, reinforcement learning, artificial neural networks.

Zusammenfassung: Die Massenpersonalisierung – ein Megatrend in der industriellen Fertigung und Produktion – erfordert eine schnelle Anpassung von Robotik- und Automatisierungslösungen an immer kleinere Losgrößen. In dieser Arbeit werden die Herausforderungen der roboterbasierten Automatisierung in einer hochgradig individualisierten Produktion aufgezeigt. Um diesen Herausforderungen zu begegnen, wird ein Framework vorgeschlagen, welches neueste Methoden des maschinellen Lernens (ML), wie beispielsweise Deep Learning, mit High-End-Physiksimulationen kombiniert. ML dient zur Programmierung und Parametrierung von Maschinen für eine bestimmte Produktionsaufgabe mit minimalem menschlichen Eingriff. Wenn die Simulationsumgebung physikalische Eigenschaften wie Kräfte oder Elastizität der realen Welt realitätsnah erfasst, stellt sie eine hochwertige Datenquelle für ML dar. Dabei werden neue Aufgaben in der Simulation schneller als in Echtzeit gemeistert, während gleichzeitig bestehende Aufgaben ausgeführt werden. Die Funktionalität des simulationsgesteuerten ML-Frameworks wird anhand eines industriellen Anwendungsfalls demonstriert.

Schlagwörter: Massenpersonalisierung, maschinelles Lernen, Simulation, Robotik, Automatisierungstechnik, verstärkendes Lernen, künstliche neuronale Netze.

1 Introduction

In the last decade, the production paradigm of "mass production" in low-mix, high-volume shifted to "mass personalization" with a high-mix, low-volume production [1]. This ongoing global paradigm shift creates challenges for manufacturing companies—especially in western high wage countries—also resulting in a massive outsourcing of production capacities to eastern low-wage countries. Increasing the degree of automation in production processes is one possible solution to fight this outflow. Mass personalization requires the automation systems to have a high degree of flexibility as well as reconfigurability to allow quick adaptation to product changes. Robot-based automation systems provide ingredients for such requirements. However, the main challenges for utilizing robotic systems in real-world mass personalization production (at a high level) are:

1. Continuously re-programming the robot to adapt to changes in production [2, 3, 4].
2. Programming and testing is time consuming and increases wear [3, 5].

***Corresponding author: Marco Huber,** Center for Cyber Cognitive Intelligence (CCI), Fraunhofer IPA, Nobelstr. 12, 70569 Stuttgart, Germany; and Institute of Industrial Manufacturing and Management IFF, University of Stuttgart, Allmandring 35, 70569 Stuttgart, Germany, e-mail: marco.huber@ieee.org, ORCID: https://orcid.org/0000-0002-8250-2092
Mohamed El-Shamouty, Kilian Kleeberger, Arik Lämmle, Fraunhofer IPA, Nobelstr. 12, 70569 Stuttgart, Germany

3. Many production tasks are complex and so is the associated robot program [6].
4. Perception of the production environment requires the derivation of information from high-dimensional data like images or three-dimensional point clouds [7].
5. ML algorithms like deep learning or reinforcement learning for overcoming the previous challenges require a sufficient amount of data [6].
6. Continuously varying safety constraints that must be maintained, particularly in human-robot collaboration (HRC) [8].
7. Solutions based on ML must be deployed to function reliably in the real-world [9].

These challenges are elaborated in more detail in Section 2. Each challenge by itself is already demanding and usually, these challenges are addressed individually. However, there is interdependence between many of these challenges and thus, a holistic approach is more promising, particularly when having real-world production systems in focus. Thus, in this paper a Hybrid Machine Learning Framework (HMLF) is proposed in Section 3 to allow addressing all challenges simultaneously. The framework combines recent advances in ML with physics simulations in order to teach robots to learn how to execute industrial tasks. Further, related work to the proposed framework is discussed. An industrial use case is presented in Section 4. It is subdivided into simplified handling, joining and collision-free point-to-point motion tasks being solved using HMLF with some first results. This paper closes with a conclusion and an outlook to future work.

2 Challenges

This section describes in detail the seven previously introduced challenges of using robots in mass personalization.

(1) Re-programming
Industrial robots are mainly utilized in repetitive scenarios for manufacturing high lot sizes to pay off the required substantial engineering efforts [2]. Programming robots is usually complex, time-consuming and requires skilled personnel [3]. Increasing changes in product models require more frequent (re-)programming of the robots by experts with process knowledge of the task at hand, as well as the integration of multiple sensors to adapt to unpredictable changes in operations [4].

(2) Time & wear
To overcome the even for specialist cumbersome process of robot programming, several tools for intuitive programming have been developed, e. g. programming by demonstration [5]. However, these approaches are usually online and therefore need the physical robot. Taking a considerable amount of time and interrupting the regular production makes intuitive online programming solutions not applicable for mass personalization [3]. This also leads to additional wear of robot components if programming and testing is performed online.

(3) Complexity
In robotics, there is the rule of thumb that tasks that are complex for a human are probably straightforward for a robot, while tasks being easy for humans tend to be highly complex for a robot. One example is the manipulation of objects with a six or seven DoF robot arm with an additional two DoF gripper. Controlling these (in total eight or nine DoF) simultaneously in a not controllable environment requires a significant amount of control engineering and many lines of code. If one considers even more challenging tasks like dexterous manipulation with a human-like robotics hand [6], it becomes apparent that at a certain point manually designing and programming robot control laws becomes almost impossible.

(4) Perception
If not relying on offline programming, where the perception of the production environment is not of relevance, it is necessary to utilize information perceived by physical sensors. Commonly employed sensor systems for environmental perception in robotics are gray-scaled, color, stereo, or time-of-flight cameras providing a two or three dimensional data representation of the environment. Based on this data, a significant amount of pre-processing, feature-engineering, pattern recognition, and algorithm design is necessary [7] in order to derive the information being relevant for accurate robot control.

(5) Data
Recent advances in ML can successfully teach robots to do complex tasks such as manipulation [6, 10, 11] and locomotion [12, 13]. Yet, generating data for ML agents on physical robots [14] or training ML agents directly on physical robots [6, 10, 11, 15] is expensive and in many cases not feasible (e. g., complex to setup the environment, wear-and-tear while generating data, etc.).

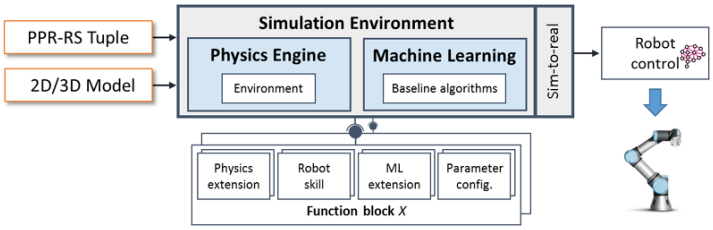


Figure 1: The proposed Hybrid Machine Learning Framework (HMLF).

(6) Safety

The application of HRC scenarios is another promising approach to increase the degree of flexibility in hybrid-automated production systems. Combining the strengths of both the human worker and the industrial robot, leverages the potentials of man and machine in industrial use cases but at the same time poses a significant effort to guarantee safety during the complete process execution. Frequent changes in the products to be manufactured also requires layout adaptations and reconfigurations. For each new layout reconfiguration, the risk assessment process has to be carried out, safety functions have to be developed and installed by the safety engineers. These time consuming processes increase the overall proportion of non-value-adding tasks, resulting in inefficient automation as the lot size decreases [8]. Overall, future automation systems need to be characterized with a high degree of autonomy, to be able to learn new behaviors, to adapt to changes without explicit programming and to guarantee safety for the human worker at all time.

(7) Deployment

When applying ML to solve parts of the aforementioned challenges, it is common to use data acquired beforehand or to rely on simulation. Thus, developing the solution is done offline. Even though the resulting solution might perform very well on the data or simulation, this is not necessarily true when the solution is deployed to the real-world production scenario. A significant amount of adaptation of the solution might be necessary to make it work properly and reliably.

3 Hybrid machine learning framework

To address the aforementioned challenges simultaneously, we propose a so-called *Hybrid Machine Learning Framework (HMLF)* as depicted in Fig. 1, where all major non-value-adding tasks are allocated to ML without taking losses in process knowledge. For this purpose, it combines three main components: physics simulator, ML, and simulation-to-reality (sim-to-real) transfer for dedicated robotics tasks. The main idea is to provide a simulation environment with an integrated physics engine, where ML is performed with simulation scenarios under realistic conditions and thus, robots learn how to solve/execute certain tasks.

Table 1: Contribution of the HMLF's main components for addressing the challenges.

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Physics simulator	✓	✓			✓		
ML: Supervised learning			✓	✓			
ML: Reinforcement learning	✓		✓			✓	
Sim-to-real							✓

As Tab. 1 indicates, combining these components allows for solving the challenges described in Section 2. In particular:

- By learning inside a simulation environment, ML can cause no harm, and the learning processes can be accelerated and parallelized based on available computational power and memory.
- Compared to standard non-physics simulator tools, using a physics engine enables the reflection of contact/collision forces, elasticity and deformability considerations, and other aspects of the real world.
- An ML framework containing different algorithms is integrated within the simulation environment to facilitate a seamless closed-loop interaction of the simulation scenarios and selected actions by the ML algorithms.
- The use of a simulation environment compensates for the lack of data and hence offers the possibility for feasible ML on virtual robots.
- ML relying on deep learning allows the extraction of patterns and information from various sensors like cameras, accelerometers, force/torque sensors, etc.

- The performance of the ML algorithms will be enhanced for dedicated tasks/processes through the encoding of expert knowledge, i. e., the current manual workers' and supervisors' experiences regarding the dependencies between process parameters, best practice rules, potential parameter variations that arise and cause problems—and what mitigation strategies they use for dealing with these variations, etc.
- To increase the transferability of the learned robot control algorithms from simulation to real world, the simulation environment is endowed with a feature for the dynamic randomization of relevant simulation parameters, e. g., position and tolerances of parts, etc., within given limits. Thus, increasing the robustness of the learned algorithms to changes and deviations in the real world.

HMLF provides a platform for engineers and ML researchers to efficiently solve highly individualized production tasks. The users of the HMLF can be divided into two main categories: *end-users* such as production managers and *developers* such as robot experts, simulation experts, machine learning experts, etc. End-users input the following items to HMLF (cf. left-hand side of Fig. 1):

- *Product-Process-Resource (PPR)* triples [16] or *PPR-Risk-Safety (PPR-RS)* tuples [17]. PPR formally links the product with the relevant resources and the production process, while PPR-RS extends PPR with the risk assessment and safety measures in HRC production scenarios.
- *2D/3D layout models* of the respective products and used resources.

Besides these inputs, the main components of the HMLF make use of and are extended by so-called *Function Blocks* (FBs, cf. bottom of Fig. 1), which are constructed by developers. FBs allow injecting domain expertise and empowering the HMLF with existing robot skills such as path planning. The following sections describe the components and how the components utilize inputs provided by the users and FBs. The framework can also be extended with function blocks for training ML models for (indirect) metrological problems, like measuring the pose or size of objects from image data as for instance required for robotic manipulation.

3.1 Physics simulators

The physics simulator is used to model the part of the real world relevant for the desired application. In case of a production process execution, it simulates the interaction of the robot and its peripherals with the environment. The simulated data is used by ML to learn/execute function blocks. In turn, the accuracy of the simulation plays a role in the accuracy of the function blocks learned. Yet, accurately simulating the interaction between different bodies is an active field of research [18] and the challenge persists to achieve the best possible compromise between accuracy of simulated interactions and performance of the simulation environment. In regards to that, the general features required from the physics engine in case of ML-driven automation are listed as follows (non-exhaustive):

1. Simulation of multi-body contacts for rigid and non-rigid bodies with accurate physical models.
2. Support of modelling robot dynamics and sensors.
3. Real-time access to simulation data.
4. Modularity or possibility of being extended with physical models.
5. Adjustable performance-accuracy trade-off of the underlying physics.

Many physics simulators applicable for robotics exist. Yet, not a single simulator outperforms all the others, especially with respect to addressing the first aforementioned feature [19]. In this paper, MuJoCo [18] and V-REP [20] are used as reference simulators. However, the framework remains generic for other physics simulators providing the aforementioned features. MuJoCo is a general-purpose physics simulator providing almost all the required features [18] with a good overall performance in robotics [19]. V-REP is a specialized simulator for robotics.

Inputs

The 2D/3D models are used for creating the environment setting in the physics simulator. In FBs, developers extend the environment to include existing robot skills, safety controllers, etc. to reflect a digital twin of the real world. Developers can also extend the physics simulator or configure the parameters of the physics engine, based on the task, e. g., for clasping objects a joining model is needed. Otherwise, the developers can ignore physics features to speed-up the simulation.

3.2 Machine learning

The ML component comprises baseline algorithms that can be used by FBs to learn how to solve given tasks. Currently, the algorithms mainly fall under two categories: Supervised learning (SL) and reinforcement learning (RL).

SL

In general, supervised ML is concerned with learning a (nonlinear) mapping $y = f(x)$ from training data $\mathcal{D} = \{(x_i, y_i), \ldots, (x_n, y_n)\}$. For the task of object detection from images, x corresponds to the image and y to the location or bounding box and the class of the objects in the image. In the HMLF, training data is generated by means of the physics simulator. Once the mapping is learned accurately enough, it can be applied on new data that can either be from the physics simulator or the real world.

While there exists a plethora of ML algorithms, HMLF mainly relies on deep learning (DL), which has shown outstanding performance particularly on image data. DL uses artificial neural networks (NNs) consisting of many hidden layers, sometimes several hundreds of them. The architecture of the NN varies depending on the task at hand. Convolutional NNs (CNNs) are chosen if image data needs to be processed, while recurrent NNs are suitable for time series data.

RL

The goal of RL is solving sequential decision making problems. The RL agent interacts with the environment, receives reward signals and accordingly learns a policy $\pi(x_t)$ that maximizes the expected reward. $\pi(x_t) = a_t$ maps from a state x_t at a given time instance t to action a_t. For instance, for an RL agent to control a robot arm to reach a (movable) target point, x_t comprises the target point and the current joint angles of a robot arm, while a_t is the motor torques of every joint provided by the agent's $\pi(x_t)$.

In order to learn the policy, RL tries to maximize the expected reward

$$Q_\pi(x_t, a_t) = \mathbb{E}[G(\tau)|\pi] \,, \tag{1}$$

where $\tau = (x_0, a_0, x_1, a_1, \ldots)$ is the state-action trajectory and $G(\tau)$ is the discounted reward of the trajectory according to

$$G(\tau) = \sum_{t=0}^{\infty} \gamma^t \cdot r(x_t, a_t) \,.$$

Here, $\gamma \in [0, 1)$ is the discount factor discounting rewards further in the future, and $r(x_t, a_t)$ is the one-step reward of being in x_t and selecting a_t according to the given policy. With this general mathematical setup, many decision-making tasks in robotics can be naturally formulated as an RL problem.

Recent advances in RL heavily rely on DL, where deep NNs are used to represent the expected reward function (1), for instance in [21].

Inputs

Developers have to divide a given task into multiple sub-tasks and for each sub-task select the best baseline algorithm to solve the respective task. For instance, Hindsight Experience Replay (HER) can efficiently solve point-to-point motion tasks without obstacles [22]. Developers also have to configure the ML parameters such as the number of NNs used in learning. Note that autonomously subdividing tasks and selecting the best algorithm and configuration to solve the respective sub-tasks is an active area of research [23] and will be considered in future work.

3.3 Sim-to-real transfer

The struggle of gathering real-world data in the domain of robotics makes simulation environments attractive for employing ML. However, ML models trained with simulation data fail when directly transferred to real robots due to the *reality gap* between the simulated and real-world environments [24].

One trivial way to tighten the gap is to increase the reality of the physics simulator by enhancing the granularity of the underlying physical models [18, 19]. However, increasing the reality of simulation comes on the cost of an increased computational demand. Other techniques address the reality gap by introducing noise in the simulation to increase the ML robustness. One technique is *domain randomization* which adds random noise to simulation parameters such as robot dynamics or sensory data [25]. Another technique is *domain adaptation* which considers the difference in data distributions when solving the same task in different domains [26]. In HMLF, domain randomization is usually employed.

Inputs

Through FBs, developers can extend the simulation models, construct sim-to-real methods and/or add collected data that can be used by the respective methods.

3.4 Related work

Existing ML frameworks provide baseline algorithms that can be used to solve general tasks [27, 28, 29]. Some approaches adapt these algorithms to solve industrial tasks on a physical robot [15, 11], which however do not scale due to the challenges aforementioned in Section 2. Other approaches cross the data hurdle by integrating ML with physics simulation [30, 22] and use sim-to-real techniques to transfer the learned task from virtual to physical

robots [24]. Our approach bases on several concepts and integrates different approaches to address the challenges mentioned in Section 2 with the focus on industrial use cases.

4 Use case

The HMLF is ought to provide solutions for a variety of production tasks. In case of industrial use cases, these tasks include grasping, manipulation, welding, assembly, packaging, quality checks, just to name a few. So far, only 10.7 % of all industrial robots in the world are used in the automation of assembly processes [31]. Though, assembly is the capstone process in manufacturing, accounting for over 50 % of the total production time and 20 % of the total production cost [32]. In the following, we therefore present the use case of electrical cabinet assembly, which has a significant potential for automation [33].

4.1 Description

Electrical cabinets are mostly customized resulting in an almost entirely manual production due to the low lot sizes. In nowadays cabinet assembly, the used components are either already available at the manual production station or the order is picked up during the work preparation. Most of the components are unwrapped and placed in boxes or bins and are therefore provided in a disordered representation to the human worker. This representation poses significant challenges to automated handling tasks.

The assembly process consists of the following steps: (1) Identifying the electrical component and its goal position in the cabinet, (2) grasping the component out of the bin, and (3) assemble it onto the DIN-Rail. Based on the selected electrical component, the joining process can be done by snap-fitting/clasping, screwing and/or riveting. The process is repeated until all components are fixed. Next, the (4) wiring of the components takes place and the (5) mounting plate is built into the cabinet followed by quality-check processes. Finally, the electrical cabinet is (6) packed and shipped to the customer. For the described use case, we assume, that the mechanical assembly is completed and the mounting plate with the DIN-Rails and cable channels is already fixed to the cabinet housing. Of interest in this paper is the process of assembling the electrical components into the electrical cabinets.

The required tasks can be summarized as handling, joining, and wiring. Out of these tasks, handling and joining electrical components have high potential for automation and hence are of interest to be learned in simulation. The wiring process is a challenging task for automation [34]. Nearly 100 % of wiring in the automotive industry is done manually [35]. Therefore, we assume that the automation solution still depends on human workers to at least carry out the wiring process in HRC. For each task class, we define a corresponding function block as follows.

4.2 Handling

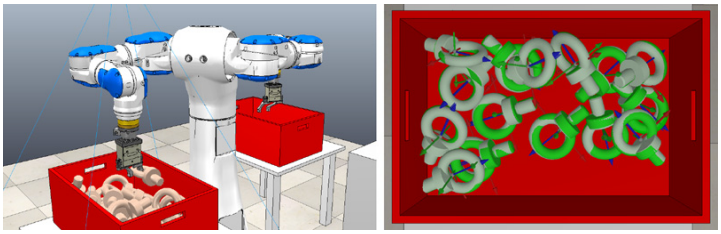
Identifying and grasping electrical components (steps 1 and 2) is challenging when they are not precisely placed in the bin and automatic handling requires a vision system for action planning. The scenario in bin-picking consists of multiple objects being chaotically stored in a bin. The task is to grasp an object out of the bin and forward it to the next production step. First, potential candidates are localized in the sensor data. Second, a suitable gripping point is selected and a collision-free path is generated. Consequently, the pose of the object relative to the gripper is known at the end and thus, the object can be precisely disposed or assembled.

Traditional (model-based) bin-picking solutions require an object-specific configuration and programming which limits the scalability. Solutions based on ML can learn to localize and grasp novel objects without requiring any manual teaching or have the potential to generalize to novel objects (model-free). The goal of the Handling Function Block (HFB) is to increase the autonomy and performance of bin-picking solutions using ML and simulation. For the transfer of the HFB from the simulation to the real world, we use domain randomization [25] instead of domain adaptation [26] because the former technique requires no samples from the real world which facilitates scalability.

Pose estimation

Knowing the pose of objects is a crucial prerequisite for many robotic grasping tasks. 6D object pose estimation (OPE) is a long-standing challenge and an open field of research since the early days of computer vision. The pose of an object is fully described by a translation vector $\mathbf{t} \in \mathbb{R}^3$ and a rotation matrix $\mathbf{R} \in SO(3)$ of its bodyfixed coordinate system relative to a given reference frame. The task of 6D OPE is challenging due to the variety of objects in the real world, potential object symmetries, clutter and occlusion in the scene and varying lighting conditions, which affect the appearance of the object in the image.

To generate scenarios typical for bin-picking, we drop objects in a random position and orientation into a bin in a



(a) (b)

Figure 2: (a) Physics simulation for data generation: Objects are dropped in a random position and orientation above a bin to generate scenes typical for bin-picking. A robot with a suitable gripper is used to simulate grasping processes. (b) Pose estimations of a deep neural network on real-world data visualized in simulation: The ground truth and the predictions are visualized in grey and green, respectively.

physics simulation as depicted in Fig. 2. Based on this, we record a depth, RGB, and segmentation image with the corresponding labels like, e. g., the object class, 6D pose, and visibility score for each object. Furthermore, the physics simulation is used to determine how well each object can be grasped in the given scenario based on physical grasping trials. For teaching new objects, no expert knowledge or time-consuming manual configuration is needed. A 3D object model is the only input that has to be provided. Based on this, a dataset is generated via simulation and is used to train a CNN for 6D OPE. To allow the model generalizing in the real world, despite being trained entirely on simulated data, various augmentations are randomly applied to the rendered training images during training, e. g., adding noise, blurring, elastic transformations, dropout, etc. The trained model is applied on a real-world dataset and achieves a high accuracy (see [36] for numerical values).[1] Fig. 2 shows some qualitative results obtained from our neural network on real-world sensor data from an Ensenso N20-1202-16-BL stereo camera.

Grasping

Grasping is a long-standing challenge in robotic manipulation. Learning vision-based robotic grasping on real-world systems can last several months and the process has to be repeated for changes in the system setup which limits scalability [14, 37]. Simulations allow reducing [26] or avoiding [25, 6] the expensive procedure of real-world data collection. A physics simulation can be used for the execution of physical grasps to, e. g., determine grasp success labels (self-supervised) for cluttered scenarios [38] or single isolated objects [39, 40].

ML is a promising approach for robotic grasping because it has the potential to generalize to novel objects that were not seen during training [14, 39, 41, 42, 43]. Those model-free methods can also be used for the HFB. In case the precise pose of a rigid object is of interest, a pose estimation step can be added to determine its pose relative to the gripper. Common model-free grasping approaches predict grasp configurations through oriented rectangles [41, 42, 43] or at pixel level [44] in the image. Other approaches like [14] and [39] sample grasp candidates and rank them using a CNN that predicts the probability of success. Using RL for robotic grasping, the robot receives a reward signal of 0 when the grasp succeeds and −1 otherwise.

4.3 Joining

In order to join an electrical component with the DIN-Rail (step 3), precise force-controlled and position-controlled robot movements are required. To simplify learning such movements, the Joining Function Block (JFB) uses already existing joining assembly skills such as in [45]. Conventionally, such skills require users to define several parameters, such as positions and the respective contact forces, in each step of execution. Steps include—assuming that the robot grasped the component in a well-defined pose—reaching a defined position above the DIN-Rail, moving linearly into contact with the cabinet or rail until reaching a defined contact force, moving linearly towards the DIN-Rail until a defined contact force, and finally, executing a rotary clasping movement while maintaining defined contact forces (cf. Fig. 3).

Manual parameterization of these skills to execute such tasks is a cumbersome process and is prone to fail in case of part or location tolerances. Several approaches use ML in learning contact rich assembly operations on phys-

1 We published a large-scale dataset for 6D OPE [36] to advance the development of novel methods with advanced ML techniques like deep learning. This dataset is used at the *"Object Pose Estimation Challenge for Bin-Picking"* at the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2019). Further information regarding the competition and the benchmark dataset are available at http://www.bin-picking.ai/.

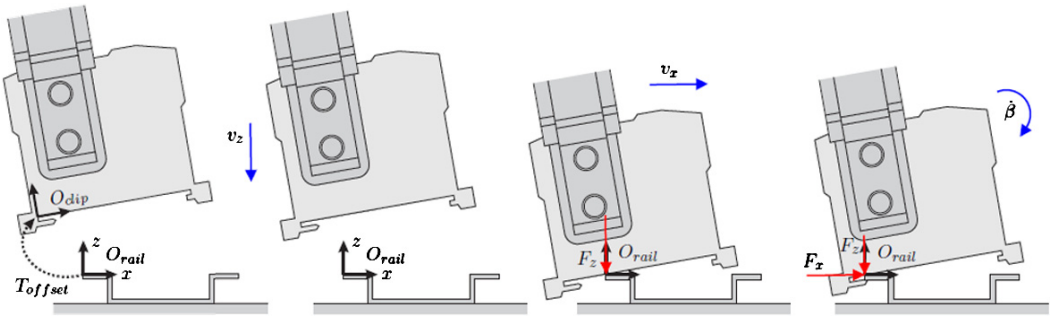


Figure 3: Steps during the automated clasping process (figure taken from [45]).

ical robots [11, 46, 2, 15]. We extend these approaches to learn in a physics simulator instead to facilitate scalability and reduce wear of robots and parts.

The goal of the JFB is to autonomously learn parameterizing the skills to successfully join the electrical components with the DIN-Rail. The JFB extends the physics simulator with joining models of the electrical components. As an input, the JFB takes additional data such as material parameters of parts of the product (e. g., modulus of elasticity, tensile strength, and cross-sectional area), process relevant parameters (e. g., maximum deviation of the snap-fits) and resource specific models (e. g., gripper and force/torque sensor). The JFB can use existing RL algorithms such as Proximal Policy Optimization [47] for reaching the goal pose with optimal deformation (e. g., not breaking the part). To make the learned algorithm robust towards tolerances, JFB uses domain randomization techniques to randomize several process relevant parameters (e. g., the location and geometry of the parts) during the learning phase. The JFB outputs a control algorithm selecting parameters and sequences of a skill-based robot program depending on recorded forces and torques applied to the robot during the process.

4.4 Safety

The wiring process (step 4) is assumed to be performed by a human worker while the robot is holding the cabinet (or the mounting plate) or executing other tasks (handling or fixation). The goal of the Safety Function Block (SFB) is to optimize solving a given task while ensuring safe interaction with the human worker and other components in the environment.

Human-robot interaction types can be classified under: isolation, co-existence, co-operation, or collaboration [48]. For each interaction-type, existing safety standards (e. g., in [49, 50]) define different safety requirements. In the isolation scenario, the robot operates in a protected workspace. The robot stops if the human worker enters the workspace. In the co-existence scenario, the robot and human worker co-exist in neighboring workspaces during operation, while in the co-operation scenario, the human worker is sharing the workspace with the robot but both are operating on different products.

In the collaboration scenario, the human worker and the robot are working on the same product simultaneously. Collaboration scenarios include complex interactions between the human worker and the robot in the 6D space. Creating 2D safety regions, which is common in industry, would constantly slow down or stop the robot due to detecting false hazards. In addition, the human worker can exhibit unexpected behaviors that can lead to a hazardous situation with the robot. This leads to strict safety requirements to ensure human's safety which on the other hand limits productivity and efficiency of the HRC scenarios. With SFB, we aim at creating an ML-based safety function that optimizes task execution using data received from sensors (e. g., multiple laser scanners, radars, etc.) and injected human behavior models in the physics simulation.

To better elaborate the SFB, we show the results of learning a simple SFB for collision-free point-to-point motion. The task is defined as reaching a given goal position in the workspace co-ordinate system (x, y, z) and safety is defined as avoiding a static obstacle located at (x_o, y_o, z_o) (cf. Fig. 4). Without the obstacle, the task can be solved by a simple inverse-kinematics solver, which is therefore included in the SFB. Accordingly, the SFB defines small steps $(\delta x, \delta y, \delta z)$ that are passed to the solver. The solver outputs the corresponding joint velocities to the robot controller. When the robot hits the obstacle, a contact force is fed back to the robot. In SFB, we define a reward function r as

$$r(x_t) = \begin{cases} 0, & \text{if the goal position is reached} \\ -20, & \text{if the robot arm hits the obstacle} \\ -1. & \text{otherwise} \end{cases}$$

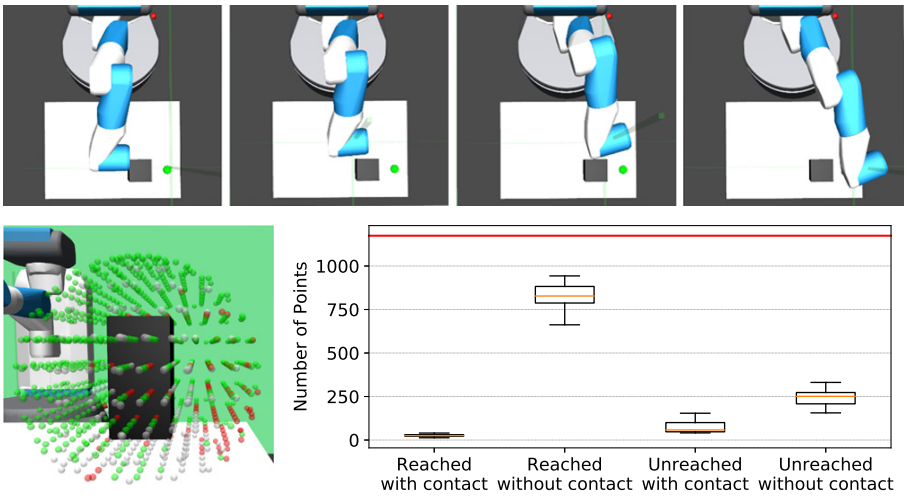


Figure 4: (Top) One validation scenario where the robot mitigates the obstacle based on the learned SFB. (Bottom-left) Validation scenarios with one obstacle position and 1,174 goals[2]. Red and green points represent goals which are successfully reached with and without collisions respectively, and gray points represent goals which are not reached. (Bottom-right) Overall performance for validation scenarios.

The baseline RL algorithm used by SFB is HER (mentioned in Section 3.2). The learning phase consists of 60k time-steps with each time-step reflecting 200 ms.

After training, the robot passes through a validation phase in which the robot is given 50 random obstacle positions, and for each obstacle position the robot has to reach 1,174 goal positions in a random sequence. Some of the goal positions lie inside the table or the obstacle to test the robot's behavior in case of infeasible goals.

Fig. 4 shows the results. On average 91.3 % (± 0.08 %) of the robot's trails to reach the goal were safe, out of which 77.2 % were successful. Out of the unreached goals, the robot tried to safely reach 77.1 % of goals but then gave up. The results show that safe behavior can be learned for static obstacles. Reaching 100 % safety is a challenging task which is devoted to future work. Still, reaching 100 % safety in the evaluations does not prove the safety of the learnt policy, specially when the learnt policy is a deep NN which is hard to interpret [51].

5 Conclusion and future work

The production paradigm of mass personalization provides a set of challenges to automation based on industrial robots. In this paper, seven major challenges have been presented. To address theses challenges, a framework was introduced that combines machine learning techniques like supervised learning and reinforcement learning with accurate physics simulations. This allows the training of robots for ever changing production scenarios on simulated data with high quality. The transfer to the real world is supported by simulation-to-reality transfer techniques. The applicability of the framework has been demonstrated on the use case of assembling electrical cabinets.

This paper introduced the concept and a first realization of the framework and demonstrated a successful transfer from simulation to the real world for object pose estimation. Further transfers to the real world are part of future work. The modular structure of the framework allows easily incorporating additional algorithms. Thus, future work is also devoted to provide a significant number of ML and sim-to-real algorithms. Furthermore, it is intended to define and implement pre-defined function blocks for many common production tasks.

Acknowledgment: The authors would like to also acknowledge Ramez Awad, Xinyang Wu, Dennis Lamaj, Jiacheng Yang and Karin Röhricht for fruitful discussions and support with implementations.

Funding: This work was partially supported by the Ministry of Economic Affairs of the state Baden-Württemberg (Zentrum für Cyber Cognitive Intelligence – Grant No. 017-192996 and Cyberprotect) and the Baden-Württemberg Stiftung (Deep Grasping – Grant No. NEU016/1).

2 Target points are uniformly distributed from the center of the obstacle (note that some infeasible goals are inside the obstacle and the table but not visible in the figure).

References

1. S. J. Hu, "Evolving paradigms of manufacturing: From mass production to mass customization and personalization", in *CMS*, Elsevier, 2013.
2. G. Thomas, M. Chien, A. Tamar, J. Ojea, P. Aparicio, and P. Abbeel, "Learning Robotic Assembly from CAD", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
3. W. Zou, M. Andulkar, and U. Berger, "Development of Robot Programming System through the use of Augmented Reality for Assembly Tasks", in *50th International Symposium on Robotics*, VDE, 2018.
4. L. Wang, B. Schmidt, M. Givehchi, and G. Adamson, "Robotic assembly planning and control with enhanced adaptability through function blocks", *The International Journal of Advanced Manufacturing Technology*, vol. 77, 2015.
5. A. Billard, S. Calinon, R. Dillmann, and S. Schaal, *Robot Programming by Demonstration*, B. Siciliano and O. Khatib (eds), Springer Berlin Heidelberg, 2008.
6. OpenAI, M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng and W. Zaremba, "Learning dexterous in-hand manipulation", 2018.
7. R. Selizki, *Computer Vision: Algorithms and Applications*, Springer, 2010.
8. F. Vicentini, M. Giussani, and L. M. Tosatti, "Trajectory-dependent safe distances in human-robot interaction", in *IEEE Emerging Technology and Factory Automation (ETFA)*, 2014.
9. S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
10. N. Fazeli, M. Oller, J. Wu, Z. Wu, J. B. Tenenbaum, and A. Rodriguez, "See, feel, act: Hierarchical learning for complex manipulation skills with multisensory fusion", *Science Robotics*, vol. 4, no. 26, 2019.
11. L. Johannsmeier, M. Gerchow, and S. Haddadin, "A framework for robot manipulation: Skill formalism, meta learning and adaptive control", *arXiv:1805.08576*, 2018.
12. S. Phaniteja, P. Dewangan, P. Guhan, A. Sarkar, and K. M. Krishna, "A deep reinforcement learning approach for dynamically stable inverse kinematics of humanoid robots", in *ROBIO*, 2017.
13. J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots", *Science Robotics*, vol. 4, no. 26, 2019.
14. S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection", in *International Symposium on Experimental Robotics (ISER)*, 2016.
15. T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control", *arXiv:1812.03201*, 2018.
16. S. Stanev, R. Awad, M. Prieur, W. Walla, and J. Ovtcharova, "Production-oriented product validation method as support for the reuse of production lines in the automotive industry", in *3rd International Conference on Changeable, Agile, Reconfigurable and Virtual Production (CARV)*, Herbert Utz Verlag, 2009.
17. R. Awad, M. Fechter, and J. van Heerden, "Integrated risk assessment and safety consideration during design of hrc workplaces", in *22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2017.
18. E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.
19. T. Erez, Y. Tassa, and E. Todorov, "Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.
20. E. Rohmer, S. P. N. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.
21. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglo, H. King, D. Kumaran, D. Wierstra, S. Legg and D. Hassabis, "Human-level control through deep reinforcement learning", *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
22. M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zarembah, "Hindsight experience replay", in *31st Conference on Neural Information Processing Systems (NIPS)*, 2017.
23. M.-A. Zöller and M. F. Huber, "Survey on Automated Machine Learning", *arXiv:1904.12054v1*, 2019.
24. X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
25. J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.
26. K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, and V. Vanhoucke, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
27. P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y. Wu, and P. Zhokhov, "Openai baselines", https://github.com/openai/baselines, 2017.
28. I. Caspi, G. Leibovich, G. Novik, and S. Endrawis, "Reinforcement learning coach", https://doi.org/10.5281/zenodo.1134899, 2017.
29. Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control", in *33rd International Conference on Machine Learning (ICML)*, vol. 48, JMLR, 2016.
30. F. Aichele, B. Schenke, B. Eckstein, and A. Groz, "A framework

for robot control software development and debugging using a real-time capable physics simulation", in *47st International Symposium on Robotics (ISR)*, 2016.
31. F. Khatami and C. Müller, "World Robotics 2018: Industrial Robots", 2018.
32. H. ElMaraghy and W. ElMaraghy, "Smart adaptable assembly systems", *CIRP*, vol. 44, 2016.
33. P. Tempel, F. Eger, A. Lechler, and A. Verl, "Schaltschrankbau 4.0: Eine Studie über die Automatisierungs- und Digitalisierungspotenziale in der Fertigung von Schaltschränken und Schaltanlagen im klassischen Maschinen- und Anlagenbau", https://discover.eplan.eu/schaltschrankbau-studie-anwender, last accessed on July 2019.
34. D. de Gregorio, R. Zanella, G. Palli, S. Pirozzi, and C. Melchiorri, "Integration of robotic vision and tactile sensing for wire-terminal insertion tasks", *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 2, 2019.
35. P. Heisler, P. Steinmetz, I. S. Yoo, and J. Franke, "Automatization of the cable-routing-process within the automated production of wiring systems", in *Energy Efficiency in Strategy of Sustainable Production III, Applied Mechanics and Materials*, Trans Tech Publications, 2017.
36. K. Kleeberger, C. Landgraf, and M. F. Huber, "Large-scale 6D Object Pose Estimation Dataset for Industrial Bin-Picking", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.
37. D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, "QT-Opt: Scalable deep reinforcement learning for vision-based robotic manipulation", in *Conference on Robot Learning (CoRL)*, 2018.
38. D. Quillen, E. Jang, O. Nachum, C. Finn, J. Ibarz, and S. Levine, "Deep reinforcement learning for vision-based robotic grasping: A simulated comparative evaluation of off-policy methods", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
39. J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-Net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics", in *Robotics: Science and Systems (RSS)*, 2017.
40. A. Depierre, E. Dellandréa, and L. Chen, "Jacquard: A large scale dataset for robotic grasp detection", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.
41. J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.
42. S. Kumra and C. Kanan, "Robotic grasp detection using deep convolutional neural networks", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.
43. I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps", in *International Journal of Robotics Research (IJRR)*, 2015.
44. D. Morrison, J. Leitner, and P. Corke, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach", in *Robotics: Science and Systems (RSS)*, 2018.
45. L. Halt, F. Nägele, P. Tenbrock, and A. Pott, "Intuitive constraint-based robot programming for robotic assembly tasks", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
46. X. Zhang, A. S. Polydoros, and J. Piater, "Learning movement assessment primitives for force interaction skills", *arXiv:1805.04354*, 2018.
47. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms", *arXiv:1707.06347*, 2017.
48. W. Bauer, M. Bender, M. Braun, P. Rally, and O. Scholtz, "Lightweight robots in manual assembly—best to start simply", *Examining companies' initial experiences with lightweight robots*, Stuttgart, 2016.
49. ISO 10218-1:2011, *Robots and robotic devices – Safety requirements for industrial robots – Part 1: Robots*, ISO, 2011.
50. ISO 10218-2:2011, *Robots and robotic devices – Safety requirements for industrial robots – Part 2: Robot systems and integration*, ISO, 2011.
51. N. Schaaf and M. F. Huber, "Enhancing Decision Tree based Interpretation of Deep Neural Networks through L1-Orthogonal Regularization", *arXiv:1904.05394*, 2019.

Bionotes

Mohamed El-Shamouty
Fraunhofer IPA, Nobelstr. 12, 70569 Stuttgart, Germany
mohamed.el-shamouty@ipa.fraunhofer.de

Mohamed El-Shamouty received his B. Sc. and M. Sc. in Communication Engineering and Media Technology from German University in Cairo, in 2015 and University of Stuttgart, in 2018 respectively. Since 2018, he is a research associate in the robotics department at the Fraunhofer Institute for Manufacturing Engineering and Automation IPA with the focus of using Machine Learning for safe and intelligent collaborative robots in production.

Kilian Kleeberger
Fraunhofer IPA, Nobelstr. 12, 70569 Stuttgart, Germany
kilian.kleeberger@ipa.fraunhofer.de

Kilian Kleeberger received his B. Sc. and M. Sc. degree in production engineering from the Friedrich-Alexander-Universität Erlangen-Nürnberg. During his studies, he focused on automation engineering, image processing and robotics. Since 2017, he is a research associate in the robotics department at the Fraunhofer Institute for Manufacturing Engineering and Automation IPA. His research focuses on computer vision and robotics.

Arik Lämmle
Fraunhofer IPA, Nobelstr. 12, 70569 Stuttgart, Germany
arik.laemmle@ipa.fraunhofer.de

Arik Lämmle received his B. Sc. and M. Sc. degree in mechanical engineering from the University of Stuttgart. During his studies, he focused on industrial robotics, production processes and machine tools. Since 2017, he is a research associate in the robotics department at the Fraunhofer Institute for Manufacturing Engineering and Automation IPA. His main focus is the development of intelligent systems for autonomous execution of assembly operations using a combination of advanced simulation environments, artificial intelligence and industrial robots.

Marco Huber
Center for Cyber Cognitive Intelligence (CCI), Fraunhofer IPA, Nobelstr. 12, 70569 Stuttgart, Germany
Institute of Industrial Manufacturing and Management IFF, University of Stuttgart, Allmandring 35, 70569 Stuttgart, Germany
marco.huber@ieee.org

Marco Huber received his diploma and Ph. D. in computer science from the Karlsruhe Institute of Technology (KIT), Germany, in 2006 and 2009, respectively. From 2009 until 2011 he was with Fraunhofer IOSB, Karlsruhe, Germany, where he was leading a research group on computer vision and information fusion. He then worked as Senior Researcher with AGT International in Darmstadt until 2015. From April 2015 to September 2018, Marco Huber was responsible for product development and data science services of the Katana division at USU Software AG in Karlsruhe. Since October 2018, he is full professor for cognitive production systems with University of Stuttgart and also head of the Center for Cyber Cognitive Intelligence (CCI) at the Fraunhofer Institute for Manufacturing Engineering and Automation IPA. His research focuses on machine learning, sensor data analysis, and robotics for production.