2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Grey-Box Learning of Adaptive Manipulation Primitives for Robotic Assembly

Marco Braun[1] and Sebastian Wrede[1]

Abstract— **Autonomous learning of robotic manipulation tasks is a promising approach to reduce manual engineering effort and increase flexibility in the future of industrial manufacturing. Although a lot of research has been done especially robotic assembly tasks requiring contact-rich compliant interaction remain a challenge for learning-based methods, since large amounts of interaction data are required. Incorporation of prior knowledge has long been seen as a possibility to make learning-based approaches tractable. The question is how can we enable process experts to encode their prior knowledge in grey-box models so that it can be used for learning robotic manipulation tasks? For that reason we propose a new grey-box learning approach, "Adaptive Manipulation Primitives" (AMP), introduced in this paper. AMPs combine compliant manipulation task specifications based on Manipulation Primitives Nets with Policy Gradient Reinforcement Learning. Our framework is evaluated in a real-world robotic assembly task. It is shown that learning to assemble industrial connector modules is possible with comparatively few real-world trials.**

I. Introduction

Nowadays, mainly position-controlled robots are used in industrial manufacturing that perform repetitive tasks in a stable environment in a fixed-programmed way [1], [2]. Assembly tasks in particular are a challenge to automate with robotic manipulators because these tasks involve contact-rich interaction. Position-controlled manipulators fail when it comes to position uncertainties [1]. Manual design of controllers that robustly perform such tasks is difficult because of the complexity of accurately modeling and estimating contact situations [2].

Classical specification based approaches rely on abstraction to manage complexity [3]–[6]. On a lower level of abstraction Manipulation Primitives (MP) describe the basic capabilities of the manipulator. The goal of a specific MP is to achieve a certain translation or rotation of the target object by applying forces and torques [7]. For compliant interaction tasks that require hybrid force and position-controlled movements, the Task Frame Formalism (TFF) has been introduced by Mason [8] to specify hybrid MPs intuitively. Bruyninckx et. al [9] and Kroeger et al. [10] have further formalized the specification of MPs based on the TFF and made them usable for practical implementation.

On a higher level of abstraction Manipulation Primitive Nets (MP-Nets) [3] or Skill Primitive Nets [11] are introduced that model the coordination of MPs based on state machines to implement the manipulation behavior. However, it is difficult to deal with position uncertainties using manual programming-based approaches. Hence, there is a need in the industry to ease the automation of complex robotic manipulation tasks that require contact-rich compliant interaction and adaption, e.g., to positional uncertainties. Manual implementation of state-dependent reactive control policies is a tedious task that usually requires a lot of fine-tuning of control parameters. Learning of control policies with Deep Reinforcement Learning (DRL) methods is a promising approach and led to successes in robot control in recent years, e.g. [2], [12], [13]. However, model-free DRL-based methods suffer from poor sample efficiency and hence are not yet applicable in industrial robotic settings [14]. To increase the sample efficiency recent approaches rely on incorporation of prior knowledge [15].

Considering manual programming-based approaches on the one hand and learning-based approaches on the other hand the question arises: how can we combine a partial behavior specification based on manipulation primitive networks with model-free RL methods to deal with uncertainties and reduce manual engineering and programming effort?

In this paper we present a grey-box learning approach that combines assembly task modeling based on MP-Nets while providing a possibility to learn control policies for selected directions of selected adaptive Manipulation Primitives. In previous work we showed the potentials to boost RL approaches in contact-rich robotic assembly processes by constraining the action space based on the TFF in a simulated Peg-in-Hole environment [19]. With this contribution the presented methodology is formally described and evaluated in real-world robotic assembly process. Our approach is termed Adaptive Manipulation Primitives (AMP). Prior knowledge is provided in form of a strategy, modeled as finite-state machine describing possible sequences of manipulation primitives. This strategy imposes constraints on the policies, but in addition provides the opportunity to learn in selected situations. By constraining the set of possible policies we aim to focus exploration on high reward regions and in that way increase the sample efficiency. A brief discussion on related work is presented in section II, followed by a detailed description of our approach in section III that is evaluated in section IV. For evaluation, AMPs are applied in a real-world industrial connector assembly task that requires dexterous manipulation skills. The paper ends with a discussion of the results in section V and an outlook and conclusion in section VI.

[1] CoR-Lab, Technical Faculty, Bielefeld University, Germany, {m.braun, sebastian.wrede}@uni-bielefeld.de

II. Related Work

Ongoing research is concerned with learning-based approaches to accomplish complex robotic manipulation tasks. The goal of learning control is to derive a policy $\pi(s)$ that maps a state vector s to a control vector, such that the desired behavior is achieved. Recent developments focus on policy search methods that are suitable for robotics because they scale with high dimensional state- and action-spaces in contrast to value-based methods [15], [20], [21].

Thomas et al. [2] present a policy search approach supported by prior knowledge for robotic assembly tasks. Properties of industrial assembly processes are used to simplify the learning problem. Usually, CAD data is available, and the geometry of the parts is known in advance. Furthermore, the environment is structured and well defined. On this basis, collision-free motion plans are calculated, which are used to compute a reward function. Their approach is based on the Guided Policy Search (GPS) algorithm of Levine et al. [22], which combines trajectory optimization and policy search. Trajectory optimization is used to find guiding patterns and avoid local optima. This approach focuses on designing rewards for movements in free space, while our approach focuses on constraints for contact-rich manipulation. [15] Dalal et al. [17] combined DRL with parameterized actions based on [18]. However, modeling the contact-rich interaction that is crucial for successful robot assembly is not covered.

As mentioned in section I in addition to learning-based methods classical approaches focus on the specification of robotic manipulation processes. The TFF introduced by Mason [8] provides an intuitive way to specify hybrid force- and position-controlled motions. Finkemeyer et al. [3] introduced Manipulation Primitive Nets that allow for the specification of complex manipulations tasks that require sensor-based control based on the TFF. Butting et al. [11] recently presented a modeling framework called Irocks to specify manipulation tasks on different levels of abstraction based on state machines that coordinate skills. This approach is very similar to the MP Net approach but focuses on usability. A taxonomy of compliant manipulation primitives for assembly tasks is proposed by Suarèz-Riuz et al. [6]. These MPs are applied in [6] to perform a Peg-in-Hole task.

Vuong et al. [16] focus on efficient learning of robotic assembly tasks, by learning sequences of predefined MPs. In that way, learning takes place on the process-level and does not cover parameters of the MPs. Recent approaches that combine Manipulation Primitive specification based on the TFF and RL methods are given by Padalkar et al. [23] and Braun et al. [15]. Braun et al. [15] focus on learning the coordination of MPs, but MPs are fixed programmed. Padalkar et al. [23] applied to learn state-dependent control policies for a single direction based on the TFF with only one MP in a vegetable cutting task, but the coordination of multiple MPs is not covered. To effectively automate complex industrial assembly tasks that require coordination of different MPs as well as the adaption of control policies, we propose to combine these approaches.

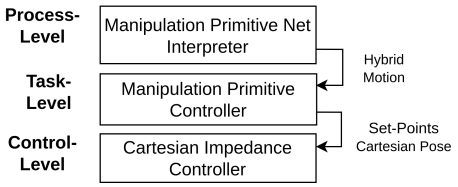


Fig. 1: Illustration of the three-layered control architecture.

III. Learning with Adaptive Manipulation Strategies

Incorporation of prior knowledge in the form of a partial strategy requires a learning architecture that allows adapting the strategy to maximize the task performance based on experience. The proposed AMP framework combines MP-Nets [3] and policy gradient RL methods. Figure 1 illustrates the control architecture of the system on the process-, task- and control-level. MP-Nets model the coordination of different MPs on the process-level. Hybrid MPs following the TFF on the task-level. A cartesian impedance controller runs on the reactive control-level. The main concepts of Adaptive Manipulation Strategies are described in detail in the following sections. MPs and MP-Nets are explained in section III-A. Section III-B is concerned with the learning of control policies for selected directions based on the TFF. Section III-B deals with the underlying learning architecture.

A. Manipulation Primitives as Basic Building Blocks for Compliant Interaction

Robotic assembly tasks can be performed by sequencing manipulation primitives (MP), which comprise the basic capabilities provided by the manipulator [6]. Finkemeyer et al. [3] describe MPs as a tuple of (1) an atomic hybrid motion HM describing a hybrid force- or position control policy for each translational and rotational direction based on the TFF [8] (2) a tool command τ and (3) a Stop-Condition λ [3]:

$$MP := \{HM, \tau, \lambda\} \tag{1}$$

Following the TFF approach [8] the control set-points for each direction (three translational direction *x, y, z* and three rotational Euler angles *a, b, c*) of the hybrid motion HM are given with respect to a task frame TF. There are three types of controllers: force/torque controller, position controller, and velocity controller. Each direction is controlled by one controller. Thereby, the type of controller can differ in all directions. E.g. the movement along the x-axis can be force controlled and simultaneously the movement along the z-axis can be position controlled. Finkemeyer et al. [3] extends the TFF by a reference frame that allows to couple the task frame to an arbitrary frame in the work cell. For one thing, the task frame can be attached to the end-effector. This setup is comparable to the classical TFF approach. For another thing, the task frame can be attached to a fixed or moving frame in the work cell, e.g., a conveyor belt that carries work pieces. During the robot motion, the task frame is adapted

accordingly. How to track the task frame during execution is defined by a tracking mode [10].

A tool command τ provides access to tool functionalities e.g. open or close a gripper or control a drill. The tool command, therefore, depends on the task-specific tool and its functionalities. A Stop-Condition λ defines the termination of an MP. It is represented by a Boolean expression that evaluates the observed sensor values S_o (as well as corresponding filter outputs) to True or False (2) [3]:

$$\lambda := S_o \rightarrow \{True, False\} \tag{2}$$

A MP runs until the associated Stop-Condition evaluates to True. Each Stop-Condition is connected to a transition that is triggered when the related Stop-Condition is met.

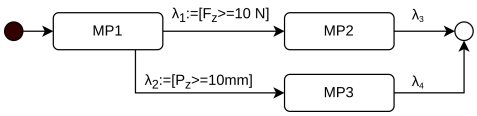


Fig. 2: Simple example of a MP-Net.

Manipulation Primitive Nets (MP-Nets) offer a higher level of abstraction by describing complex behavior as coordination of MPs modeled as a state chart (see [3]). A simple example of an MP-Net is shown in Figure 2. Initially, MP1 is executed until λ_1 or λ_2 evaluates to true. Subsequently, control is handed over to MP2 in case of λ_1 or MP3 in case of λ_2, respectively. Butting et al. [11] have recently described a model-based approach to specify MP nets (they call it Skill Primitive Nets), which is also based on the graphical state diagram language and follows a similar approach. For a more detailed description of MPs and MP-Nets, we refer to [3] and [10].

B. Adaptive Manipulation Primitives

Following the classical TFF approach, position-, velocity- or force-control set-points are manually specified to fixed values. This concept is extended to include learning capabilities by providing the ability to adapt controller set-points state-dependently via reinforcement learning methods. The Reinforcement Learning framework provides a possibility to learn from interaction with the environment. A robotic assembly task can be formalized as a Markov Decision Process (MDP) that is a 4-tuple, (S, A, T, R). S is a set of states, A is a set of actions, T are transition probabilities mapping S x A x S, and R is a reward function [24]. During execution of an adaptive Manipulation Primitive set-points w_d for selected agent controlled directions d are sampled from a state s dependent learned policy $\pi(s)$.

$$w_d \sim \pi(s), s \in S \tag{3}$$

Instead of a fixed set-point a continuous range is specified which constraints the policy output. The policy that outputs the actual state-dependent control set-point is learned as described in the following section. Especially tasks, that evolve a lot of manual fine-tuning if they are programmed in a fixed way, like the assembly of tight-fitted parts in the presence of uncertainties.

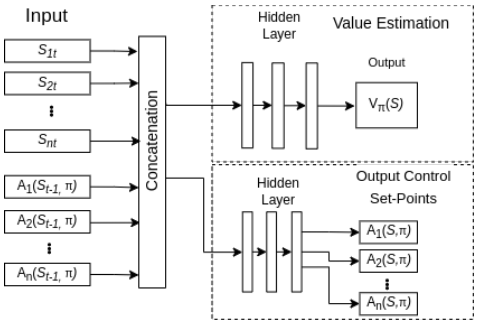


Fig. 3: Schematic of the actor-critic learning architecture consisting of one value network and one policy network.

The key to learning is a reward signal that encodes the task objectives. Policy search methods aim to optimize directly a policy that maps state inputs to actions. This is achieved by adapting the policy to maximize the sum of all rewards per episode. We refer to a concise description of learning robotic manipulation tasks given by [15]. In the following, a policy gradient RL method to learn based on AMP is introduced.

Figure 3 shows the actor-critic learning architecture including a policy network and a value network. When executing an adaptive MP with n agent-controlled directions, the corresponding action space is n-dimensional. Each action dimension A_n corresponds to one agent-controlled direction d. For each adaptive MP, a differentiable representation of the policy $\pi(a|s, \theta)$ in the form of an artificial neural network is used that outputs depending on weights vector $\theta \in \mathbb{R}^d$ and the state input S_t, with S_t is the state information at time t. To give access to history S_t is a concatenation of the available sensor data S_o at time t and the last action vector $A(S_{t-1}, \pi)$ at time $t-1$ is fed into the network. A value network with weights vector ρ is applied to estimate the future Return G_t (4) that is defined as the sum of discounted future rewards R_t being in state S_t and following the learned policy (5) [25]:

$$G_t \doteq \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k \tag{4}$$

$$V_\pi(s, \rho) \doteq \mathbb{E}_\pi[G_t | S_t = s] \tag{5}$$

Concerning AMP, the value network predicts the future return G_t when sampling the controller set-points from the policy. These value estimates $V(S_t)$ are used to update the policy of the adaptive MP. Updates of the value network are performed every time an adaptive MP is executed (6) [25, p. 120]:

$$V(S_t) \leftarrow V(S_t) + \alpha[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)] \tag{6}$$

This collected reward and the estimated future reward are used to adapt the policy of each adaptive MP by an

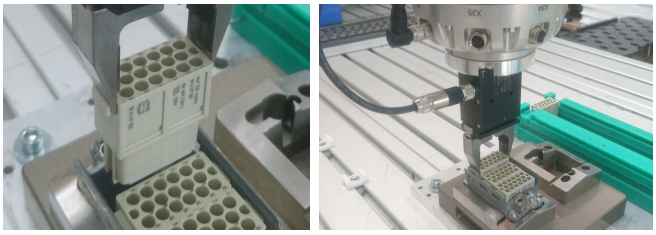


Fig. 4: Robotic assembly of industrial connector modules.

actor-critic update following the Proximal Policy Optimization (PPO) method [26]. PPO was selected because it has been successfully used to learn skillful policies for robot manipulation tasks [12]. Although PPO would allow for fully incremental learning, optimization with mini-batches usually reaches better performance [26]. Therefore, the policy is run for several timesteps, collecting the estimated advantages, and afterward, the policy is updated with mini-batches.

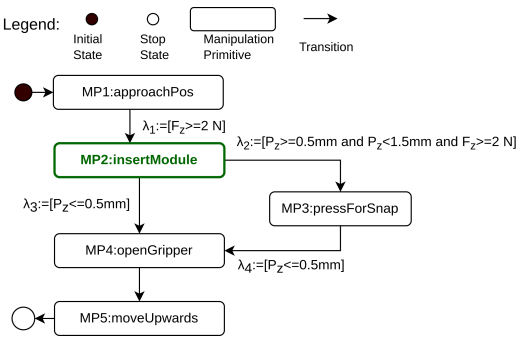
IV. Evaluation

We evaluate the applicability of AMP in a real-world industrial assembly task. First, the task is described in more detail, followed by the formalization as MDP. Afterward, experimental results are presented. The Han-Modular industrial connectors from the manufacturer HARTING consists of multiple modules with different mating surfaces that need to be inserted into a frame in the production process as depicted in Figure 4. The task is representative for a typical assembly task in industry. This is a variation of a Peg-in-Hole insertion task with only 0.2-0.4 mm clearance but with different geometric shapes of the parts (see Figure 6). Assembly of tight-fitted parts like these is a challenging task for robotic manipulators. High precision and dexterous manipulation skills are required for robust execution. Manual programming of a robust insertion behavior that can cope with uncertainties as well as different geometric shapes would require a lot of fine-tuning and becomes very complex [**?**], [27]. In practice, especially the last module is hard to insert with manually programmed behavior because already mounted modules must be skillfully pushed aside to create enough space. With AMPs we are able to learn the insertion process of different modules under positional uncertainties with the same underlying model that is described in the following.

A. Description of the Han-Modular Assembly

Figure 5 shows an excerpt of the Han-Modular Assembly AMPs that model the insertion sub-process. Corresponding control configurations of the applied Manipulation Primitives are listed in the table below.

In the initial configuration, the end-effector is positioned approximately above the target position with the module already gripped. Entering the excerpt of the AMP, the robot approaches the frame executing *MP1:approach* by moving downwards with 10mm/s in z-direction, while all other directions are position-controlled. When a contact force of

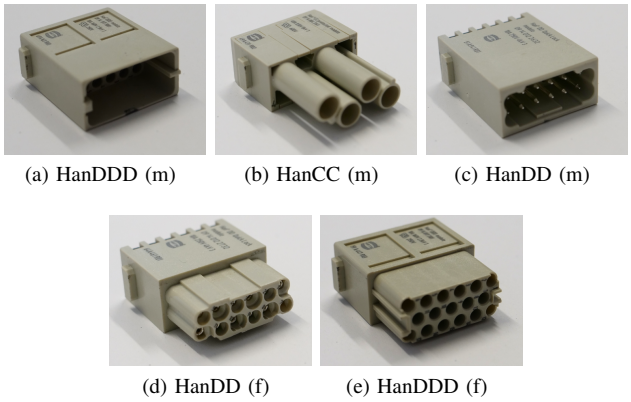


ID	x	y	z	a	b	c
MP1	0 mm	0 mm	-10 mm/s	0 rd	0 rd	0 rd
MP2	$v_x(\pi, s)$	$v_y(\pi, s)$	2 N	0 rd	0 rd	0 rd
MP3	0 N	0 N	-1 mm/s	0 rd	0 rd	0 rd
MP4	0 m/s	0 m/s	0 m/s	0 rd	0 rd	0 rd
MP5	0 N	0 N	2 mm/s	0 rd	0 rd	0 rd

Fig. 5: AMP describing the insertion policy for Han-Modular insertion. The controller set-points for each direction of the MPs are shown in the table below the diagram. Adaptive MPs and learned set-points are colored green.

2 N is measured, the adaptive MP2 is executed, performing the actual insertion step. Thereby, the orientation of the end-effector is fixed. Contact between the module and the frame is enforced by applying 2 N in z-direction, while the robot moves in x- and y-direction with set-points sampled from the learned policy $\pi(s_t)$. Subsequently, there are two possible Stop-Conditions. *MP4:openGripper* is executed, when the module is already fully inserted. *MP3:pressForSnap* is executed, when the module is almost fully inserted (0.5 mm<distance<1.5 mm) but needs to be pressed into the frame for the last mm in z-direction to snap-in. x- and y-direction are controlled to 0 N making the end-effector freely movable in this directions. When the module is successfully inserted, the execution continues with *MP4:openGripper*. Finally, *MP5:moveUpwards* is executed, and the end-effector moves above the frame. A Kuka LBR iiwa robot is used to execute the hybrid motions in cartesian-impedance mode. This robot provides force and pose information of the shelf, therefore no additional force-torque-sensor is needed. A parallel gripper with custom made gripper jaws to handle the Han modules is attached to the end-effector.

We selected a Han DDD male module as shown in Figure 6 a) for initial evaluation of our approach. Assembly of the Han DDD male module is challenging because the shape fits tightly, and there is no bevel at the front edge that eases the insertion. The maximum forces in x- and y-direction are limited to 4 N to prevent damage to the manipulated parts. To learn the insertion behavior, the execution time of MP2 is fixed in the training phase to 30 time steps at 10 Hz, which corresponds to 3 s duration for one training episode. At a rate of 10 Hz, the observations are collected, the reward is calculated based on the distance to the target position, the control set-points for the x and y directions



(a) HanDDD (m) (b) HanCC (m) (c) HanDD (m)

(d) HanDD (f) (e) HanDDD (f)

Fig. 6: Different modules of the Han-Modular assortment.

controlled by the agent are sampled from the policy, and finally the hybrid motion is executed until the start of the next cycle. As mentioned in the introduction position uncertainties are one reason, why manual programming-based approaches are hard to accomplish. To simulate pose estimation errors and resulting position uncertainties the approaching pose is superimposed with noise in *x*- and *y*-direction drawn from a normal distribution with standard deviation $\sigma_{x,y} = 1mm$ in every episode. In this way, learning of a robust policy that can deal with position uncertainties is enforced.

B. Modeling the Insertion of Connector Modules as MDP

State Space S: Actions are selected based on a six-dimensional state vector: (1) force at the end-effector in *x*-direction, (2) force at the end-effector in *y*-direction, (3) force at the end-effector in *z*-direction, (4) the position of the end-effector in *z*-direction, (5) and (6) the last action in *x*-direction and *y*-direction, respectively (see III-B).

Reward function r(t): The reward r_t is calculated anti-proportional to the distance between the actual module pose and the target pose in *Z*-direction (7):

$$r(t) = -(z_{module}(t) - z_{target}) \tag{7}$$

Maximizing the sum of rewards is achieved by minimizing the distance in *z*-direction. There is no bonus for successful insertion. Since the execution of MP2 is fixed at 3 s, the reward implicitly reinforces fast insertion behavior as the sum of rewards increases across the episode.

Action Space A: During the execution of *MP2:insertModule*, the end-effector velocity set-points in x- and y-direction are sampled from the state S dependent learned policy π. The policy output is a two-dimensional action vector A:

$$A = \begin{bmatrix} v_x(\pi, s) \\ v_y(\pi, s) \end{bmatrix} \quad v_x(\pi, s), v_y(\pi, s) \in [-10, 10]mm/s \tag{8}$$

For the module insertion task, the policy output is limited to $-10mm/s$ to $10mm/s$.

Policy Representation and Learning: An ANN with two hidden layers of 256 neurons each is used to represent the policy. Policy updates follow the PPO algorithm described in section III-B. To learn the insertion policy we use RLlib [28]. RLlib is an open-source library for Reinforcement Learning that provides software primitives [28] and also complete algorithm implementations. RLlib provides a PPO implementation of the shelf. RLlib uses open ai gym [29] as environment interface. For the adaptive *MP2:insertModule* a custom gym environment is implemented. For learning the policy of *MP2:insertModule* with the following hyper-parameters are applied: (1) Optimizer = Adam, (2) learning rate = $5e^{-5}$, (3) normalize observations = True, (4) importance ratio clipping = 0.2, (5) normalize rewards = True, (6) batchsize=150.

C. Learning Results

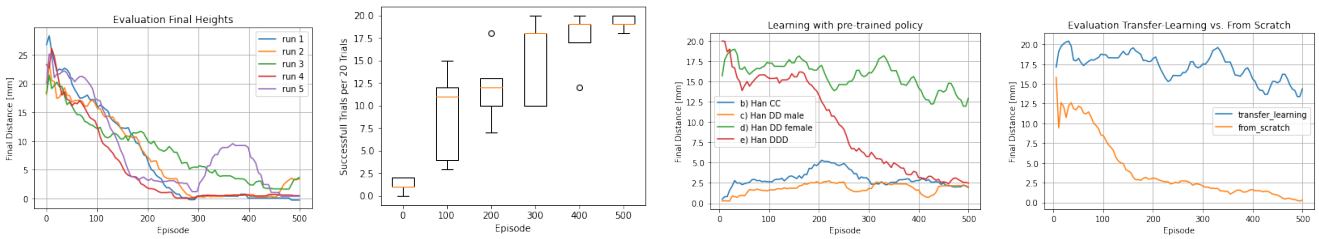
The learning progress of five independent learners is depicted in Figure 7a. Five learners start with the same initial configuration. During the training phase, the applied PPO algorithm updates the policy to reach a maximum return as described in section III-B. The final distance of the inserted module to the target position in *z*-direction at the end of each learning episode, as shown in Figure 7a decreases continuously until about 300 learning episodes. From 300 to 500 episodes, the increase in performance slows down, and one learner shows a performance drop but catches up afterward. A video showing the execution of the learned AMP is available online.[1]

We define a successful policy rollout when the distance in *z*-direction between the actual pose and target pose does not exceed the minimum possible distance by more than 0.5 mm to measure the task performance. Every 100 learning episodes, the performance of the policy is evaluated. The success rate of five independent learners is depicted in Figure 7b. At the beginning of the training phase, low success rates are achieved by random exploration. Median performance increases in the first 100 learning episodes, but there are significant differences between the best (15/20 success rate) and the worst learner (3/20 success rate). After 300 learning episodes, a first learner achieves 20/20 successful insertions, but poorer learners achieve only 10/20 successful attempts. After 500 learning episodes, which corresponds to an interaction time of 40 minutes, a success rate of about 95% is achieved. The worst learner achieves 18/20 successful attempts, and the median is about 19/20 successful attempts.

D. Generalization and Transfer Learning

The Han modular assortment includes a variety of modules with different mating surfaces and geometric shapes. Ideally, a learned policy is able to generalize over the different geometric shapes. Therefore, we analyze the generalization capabilities of the initially learned policy with another four different shaped modules that are depicted in Figure 6. Figure 7c shows the moving average of the final distance from the goal position with the pre-trained policy. The performance depends on the concrete module shape. Module b)

[1] https://uni-bielefeld.sciebo.de/s/Gkf6tVheLu3F9KQ



(a) Final episode heights. (b) Rate of successful insertions. (c) Generalization of different modules. (d) From scratch vs. transfer.

Fig. 7: Evaluation results of the industrial connector module insertion task.

HanCC and c) HanDD male immediately reach high success rates, but the performance decreases for about 200 training episodes until it increases again. Modules (d) HanDD female and (e) HanDD initially perform poorly with the pre-trained policy, but the performance of module e) increases fast, whereas the performance of module d) only increases slowly. During the experiment, we observed that the insertion of module d) gets stuck very often the half way, when the edge in the middle of the module hits the frame. This configuration forms a local optimum that did not exist in the original training task. Since the insertion of module d) performs poorly following the transfer learning approach, we compare the performance of transfer learning and learning from scratch as shown in Figure 7d. Learning progress is much faster when learning from scratch than learning with a pre-trained policy trained with the reference module a).

V. Discussion

The presented grey-box learning approach has been successfully applied in a real-world robotic assembly task. Especially, dealing with position uncertainties and the variance of geometric shapes of the assembled parts is feasible without the need to describe every detail of the robot's behavior. By constraining the possible behaviors, the provided AMP model facilitates the learning problem, and learning the insertion policy was enabled with an applicable number of trials in the real world.

The main advantage of our approach is that the range of possible AMP models extends from exclusively manual programming to full learning. On the one hand, constraining the robot behavior can lead to sub-optimal solutions, on the other hand, learning with an unconstrained action space is not applicable for manufacturing and not efficient. The trade-off between modeling the behavior and learning selected degrees of freedom must be made by the process expert, which is considered to be a challenging and complex task. In addition to specifying the hybrid motion, the process expert needs to possess an understanding of the underlying RL algorithm. However, we believe that providing use cases such as the one presented, which mainly requires slight customization, would reduce the complexity in the design phase. In the investigated connector assembly scenario, we only modeled one adaptive MP to learn the skillful insertion of the components with only two RL-controlled directions, while upstream and downstream process steps are fully specified. Such an approach can be applied to many other assembly processes where positional uncertainties and variance in geometric shapes cause a complete manual specification to become very complex. In industry, for example, there is very often the application case that parts are gripped based on camera images, from bulk material or from the conveyor belt, and must be inserted very precisely for further processing.

VI. Conclusion and Future Work

In this paper we introduced a grey-box learning approach that combines assembly task modeling based on MP-Nets with the possibility to learn control policies for selected directions in selected adaptive Manipulation Primitives. Whereas many approaches focus on reward shaping our approach imposes constraints on the policy by providing a partial behavior specification. In this way, not arbitrary movements but just those that comply with the AMP model can be performed. However, reward shaping and constraining the policy are not mutually exclusive and can be combined [30]. Constraining the policy space is a promising approach regarding complex manipulation tasks in particular robotic assembly.

As described in section IV the achieved sample efficiency is sufficient to learn the industrial connector module insertion task within 500 episodes, which corresponds to only 40 minutes of interaction time. One possibility to further increase the sample efficiency would be off-policy learning and the usage of experience replay buffers as presented by [31].

The transfer learning approach, as described in section IV-D, is less sample efficient compared to learning from scratch. One possible reason is over-fitting of the policy model to the specific geometric shape of one module. An analysis of the extent to which, for example, other network architectures can improve transfer learning remains future work.

Although the basic methodology is robot agnostic, we have so far implemented and tested MP control only for the Kuka LBR iiwa robot. However, to support classical position-controlled industrial manipulators, a replacement of the cartesian-impedance controller by an admittance-control [32] approach is possible and should be investigated further.

REFERENCES

[1] Z. Zhu, "Robot Learning from Demonstration in Robotic Assembly: A Survey," *Robotics*, vol. 7, no. 2, p. 17, 2018.

[2] G. Thomas, M. Chien, A. Tamar, J. A. Ojea, and P. Abbeel, "Learning Robotic Assembly from CAD," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2018, pp. 3524–3531. [Online]. Available: http://arxiv.org/abs/1803.07635https://ieeexplore.ieee.org/document/8460696/

[3] B. Finkemeyer, T. Kröger, and F. M. Wahl, "Executing assembly tasks specified by manipulation primitive nets," *Advanced Robotics*, vol. 19, no. 5, pp. 591–611, 2005.

[4] J. Malec, K. Nilsson, and H. Bruyninckx, "Describing assembly tasks in a declarative way," *ICRA 2013 WS on semantics, identification and control of robot-human-environment interaction*, pp. 1–4, 2013.

[5] J. Felip, J. Laaksonen, A. Morales, and V. Kyrki, "Manipulation primitives: A paradigm for abstraction and execution of grasping and manipulation tasks," *Robotics and Autonomous Systems*, vol. 61, no. 3, pp. 283–296, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2012.11.010

[6] F. Suárez-Ruiz and Q. C. Pham, "A framework for fine robotic assembly," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2016-June, pp. 421–426, 2016.

[7] A. Morales, M. Prats, P. Sanz, and A. P. Pobil, "An Experiment in the Use of Manipulation Primitives and Tactile Perception for Reactive Grasping," *October*, pp. 1–7, 2007.

[8] M. T. Mason, "Compliance and Force Control for Computer Controlled Manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, no. 6, pp. 418–432, 1981. [Online]. Available: http://ieeexplore.ieee.org/document/4308708/

[9] H. Bruyninckx and J. De Schutter, "Specification of force-controlled actions in the "task frame formalism"-a synthesis," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 581–589, 1996. [Online]. Available: http://ieeexplore.ieee.org/document/508440/

[10] T. Kroeger, B. Finkemeyer, and F. Wahl, "A task frame formalism for practical implementations," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 2004, no. 5. IEEE, 2004, pp. 5218–5223. [Online]. Available: http://ieeexplore.ieee.org/document/1302546/

[11] A. Butting, B. Rumpe, C. Schulze, U. Thomas, and A. Wortmann, "Modeling Reusable, Platform-Independent Robot Assembly Processes," no. DSLRob, jan 2016. [Online]. Available: http://arxiv.org/abs/1601.02452

[12] 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," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, aug 2018. [Online]. Available: http://arxiv.org/abs/1808.00177

[13] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection," mar 2016. [Online]. Available: http://arxiv.org/abs/1603.02199

[14] G. Schoettler, A. Nair, J. Luo, S. Bahl, J. Aparicio Ojea, E. Solowjow, and S. Levine, "Deep Reinforcement Learning for Industrial Insertion Tasks with Visual Inputs and Natural Rewards," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, oct 2020, pp. 5548–5555. [Online]. Available: http://arxiv.org/abs/1906.05841https://ieeexplore.ieee.org/document/9341714/

[15] M. Braun and S. Wrede, "Incorporation of Expert Knowledge for Learning Robotic Assembly Tasks," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 2020-Septe. IEEE, sep 2020, pp. 1594–1601. [Online]. Available: https://ieeexplore.ieee.org/document/9211917/

[16] N. Vuong, H. Pham, and Q.-C. Pham, "Learning Sequences of Manipulation Primitives for Robotic Assembly," nov 2020. [Online]. Available: http://arxiv.org/abs/2011.00778

[17] M. Dalal, D. Pathak, and R. Salakhutdinov, "Accelerating Robotic Reinforcement Learning via Parameterized Action Primitives," *Advances in Neural Information Processing Systems*, vol. 26, no. NeurIPS, pp. 21 847–21 859, oct 2021. [Online]. Available: http://arxiv.org/abs/2110.15360

[18] W. Masson, P. Ranchod, and G. Konidaris, "Reinforcement Learning with Parameterized Actions," *30th AAAI Conference on Artificial Intelligence, AAAI 2016*, pp. 1934–1940, sep 2015. [Online]. Available: http://arxiv.org/abs/1509.01644

[19] M. Braun and S. Wrede, "Boosting reinforcement learning of robotic assembly tasks by constraining the actionspace in a Task-Specific manner," in *54th International Symposium on Robotics (ISR Europe 2022) (ISR Europe 2022)*, Munich, Germany, Jun. 2022.

[20] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, pp. 156–163, 2015.

[21] S. Schaal and C. G. Atkeson, "Learning Control in Robotics," *IEEE Robotics and Automation Magazine*, vol. 17, no. 2, pp. 20–29, jun 2010. [Online]. Available: https://ieeexplore.ieee.org/document/5480446/

[22] S. Levine and V. Koltun, "Guided Policy Search," *ICML'13 Proceedings of the 30th International Conference on International Conference on Machine Learning*, vol. 28, 2013.

[23] A. Padalkar, M. Nieuwenhuisen, S. Schneider, and D. Schulz, "Learning to Close the Gap: Combining Task Frame Formalism and Reinforcement Learning for Compliant Vegetable Cutting," in *Proceedings of the 17th International Conference on Informatics in Control, Automation and Robotics*, no. Icinco. SCITEPRESS - Science and Technology Publications, 2020, pp. 221–231.

[24] R. Parr and S. Russell, "Reinforcement learning with hierarchies of machines," *Neural Information Processing Systems (NIPS)*, pp. 1043–1049, 1998.

[25] A. G. Sutton, R. S. and Barto, *Reinforcement Learning: An Introduction (2nd Edition, in preparation)*, 2018.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," pp. 1–12, 2017. [Online]. Available: http://arxiv.org/abs/1707.06347

[27] S. R. Chhatpar and M. S. Branicky, "Search strategies for peg-in-hole assemblies with position uncertainty," *IEEE International Conference on Intelligent Robots and Systems*, vol. 3, pp. 1465–1470, 2001.

[28] E. Liang, R. Liaw, P. Moritz, R. Nishihara, R. Fox, K. Goldberg, J. E. Gonzalez, M. I. Jordan, and I. Stoica, "RLlib: Abstractions for Distributed Reinforcement Learning," *35th International Conference on Machine Learning, ICML 2018*, vol. 7, pp. 4768–4780, dec 2017. [Online]. Available: http://arxiv.org/abs/1712.09381

[29] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.

[30] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained Policy Optimization," 2017. [Online]. Available: http://arxiv.org/abs/1705.10528

[31] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas, "Sample Efficient Actor-Critic with Experience Replay," *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, no. 2016, nov 2016. [Online]. Available: http://arxiv.org/abs/1611.01224

[32] M. Schumacher, J. Wojtusch, P. Beckerle, and O. von Stryk, "An introductory review of active compliant control," *Robotics and Autonomous Systems*, vol. 119, no. July, pp. 185–200, 2019.