AI MOTION CONTROL – A GENERIC APPROACH TO DEVELOP CONTROL POLICIES FOR ROBOTIC MANIPULATION TASKS

Kurrek, Philip (1); Jocas, Mark (1); Zoghlami, Firas (1); Stoelen, Martin (2); Salehi, Vahid (1)

1: University of Applied Sciences Munich; 2: University of Plymouth

ABSTRACT

Current robotic solutions are able to manage specialized tasks, but they cannot perform intelligent actions which are based on experience. Autonomous robots that are able to succeed in complex environments like production plants need the ability to customize their capabilities. With the usage of artificial intelligence (AI) it is possible to train robot control policies without explicitly programming how to achieve desired goals. We introduce AI Motion Control (AIMC) a generic approach to develop control policies for diverse robots, environments and manipulation tasks. For safety reasons, but also to save investments and development time, motion control policies can first be trained in simulation and then transferred to real applications. This work uses the descriptive study I according to Blessing and Chakrabarti and is about the identification of this research gap. We combine latest motion control and reinforcement learning results and show the potential of AIMC for robotic technologies with industrial use cases.

Keywords: Design methods, Artificial intelligence, Motion control, Robotics, Machine learning

Contact:
Kurrek, Philip
Munich University of Applied Sciences
Department of Applied Sciences and Mechatronics
Germany
philip.kurrek@hm.edu

Cite this article: Kurrek, P., Jocas, M., Zoghlami, F., Stoelen, M., Salehi, V. (2019) 'Ai Motion Control – A Generic Approach to Develop Control Policies for Robotic Manipulation Tasks', in *Proceedings of the 22nd International Conference on Engineering Design (ICED19),* Delft, The Netherlands, 5-8 August 2019. DOI:10.1017/dsi.2019.363

1 INTRODUCTION

The increase in product individualization strengthens the importance of decoupled factories and the flexibility in automation and manufacturing. The European Group on Ethics in Science and New Technologies published an article with five relevant developments for the industry: Artificial intelligence in the form of machine learning, advanced mechatronics, systems which can perform tasks independently from human operators, systems which are fully autonomous and systems which are able to interact between humans or other machines (EuropeanCommission, 2018). These developments give robots new ways to generate control policies for manufacturing tasks in production. Contemporary controllers are mostly developed for stationary systems safeguarded by cages, for an environment where humans operate from control centers, outside the aforementioned cages, or behind safety walls (Pfeiffer, 2016). The Federation of Robotics estimates that from 2017 to 2021, the supply of industrial robots will increase from 318,000 to 630,000 units. The large number of robots estimated is due to a combination of the robots' inability to perform multiple varied tasks, and the large quantity of specialized tasks need to be managed by the robots. (IFR, 2018)

Autonomous robots are classified according to ten levels of autonomy for unmanned systems. In comparison, to the five autonomous level of self-driving vehicles, there a three factors which categorize the autonomy ability (Figure 1). The autonomy of a robot is depending on human independence, the mission and the environmental complexity. Thus, a robot which is capable of working completely autonomously in an environment with low complexity is not classified under autonomy level 10 (Beer, 2014). Currently, robots in high complexity environments require Human-Robot-Interaction (HRI) to succeed their tasks, because most of these robots do not have intelligent controllers which can adapt to different conditions.

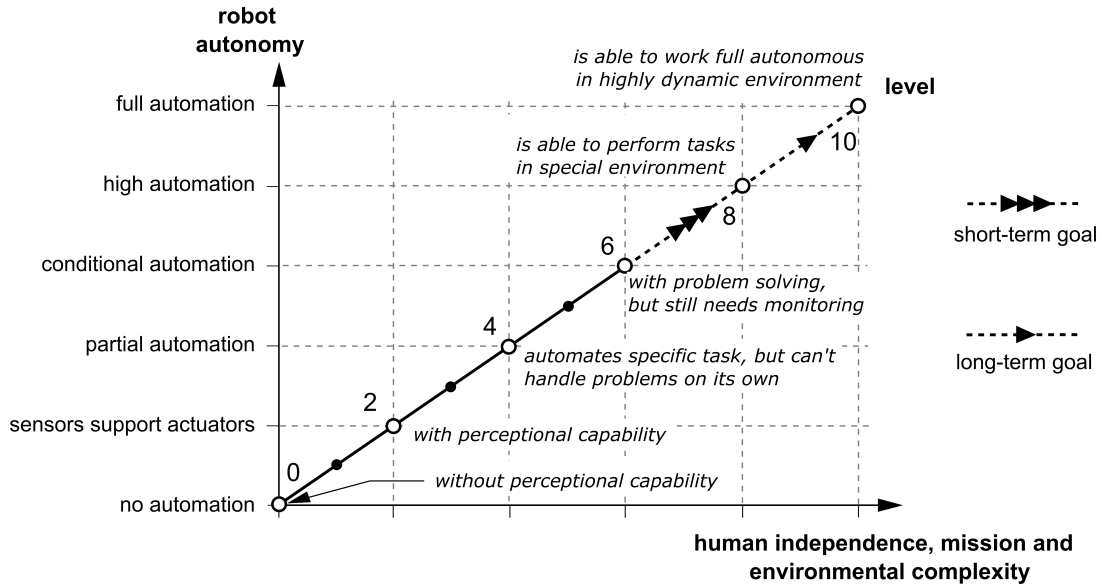

Figure 1. Robot autonomy based on ALFUS, the autonomy levels for unmanned systems (Beer, 2014).

The second machine age describes a future in which robots achieve the intelligence to perform cognitive tasks on par with humans (Wisskirchen, 2017). It is difficult to determine the exact level of autonomy of a system, but we estimate that modern technologically advanced robots have an autonomy level between 4 and 6. If fitted with an intelligent motion controller, robots should be able to achieve a level of 8 as a short term goal. Artificial Intelligence (AI) can be used to enhance the capabilities of a system, which is complicated to achieve with traditional methods. We introduce AI Motion Control, or "AIMC" for short, which is a generic approach to develop control policies for a dynamic context. AIMC also describes a shorter way to develop robot motion controllers, while nevertheless achieving similar manipulation tasks and trajectories. This research presents the identification of the research gap and gives an outlook of the development of this approach (Blessing and Chakrabarti, 2009).

https://doi.org/10.1017/dsi.2019.363 Published online by Cambridge University Press

2 RELATED WORK

The new technological challenges faced by the robots have created a need for new methods for the development of the solutions for these challenges. A supervisor can teach the robot, perception can support with visual input to train a controller or machine learning can allow the robot to learn control policies through experience. This section gives an overview of intelligent motion controllers, which require low adaptation to generate manipulation tasks.

Model learning for robot control presents how to use forward and inverse models, how to mix these models and how to fit the motion control policy through machine learning (Nguyen-Tuong and Peters, 2011). With learning from demonstration or imitation, which is combined in Imitation Learning, it is possible to avoid manually hard coding control policies and to transfer human knowledge to an agent. Machine learning techniques can be used to build a generic model of robot movements from a few examples provided by a supervisor (Zadeh, 2012). In another work, a robot learns trajectories for a desired motion by using refitted linear models and then unifies these trajectories into a single control policy, this approach is named Guided Policy Search (Levine1 *et al*., 2015). For instance, perception guided motion controllers use visuo-tactile information to manipulate and explore unknown objects (Li *et al*., 2015). A framework saves perceptual feedback in a visuomotor memory and learns from demonstration. Recurrent neural networks can also be used to analyze these information (Sasaki *et al*., 2016). Image processing can be powerful, but for some use cases it is essential to understand three-dimensional models. A multi-model approach can analyze 3D functional features of tools (Mar *et al*., 2015). It has given a robot the ability to learn how to screw a cap onto a bottle through the combination of extracting visuomotor information and the training of machine learning algorithm (Levine *et al*., 2015). The deep visuomotor policies were trained using reinforcement learning (RL). RL can be also used to train predictive policies for skilled object grasping and ball throwing (Ghadirzadeh *et al*., 2017). Most of the RL approaches use simulations to visualize the learning problem in order to plan which function works best. Simulations are also used to improve the efficiency of Deep Robotic Grasping. For example, a given level of performance achieved by randomly generated simulated objects can reduce the real-world data needed (Bousmalis *et al*., 2017). It is also proven that through modularity, there can be a transfer of control policies from simulation to reality (Clavera *et al*., 2017).

3 PROBLEM DESCRIPTION

The presented approaches combine AI with motion control. For example, computer vision offers robots the ability to calculate gripping points or avoid collision. Other use cases exist in which observations cannot give the robots enough information to guarantee the successful manipulation of objects. The presented approaches demonstrate how to generate intelligent motion control for manipulation tasks, mostly in laboratories under simplified conditions and for specialized robotic tasks. Through reinforcement learning, robots are able to learn trajectories through trial-and-error based on experience. This paper proposes an answer to the following research question: Is it possible to develop multiple robot manipulation tasks without explicitly programming how to achieve desired goals? We also show what information is needed to design such a system, what the subsystems of an intelligent motion control are, how can they be combined and which advantages will a robot have by using intelligent motion control. We compare the development of motion control policies using the proposed approach with traditional development approaches. This work introduces an approach which contains the application of AI methods in a robot motion control development. As a result, the development becomes more flexible and scalable, which is necessary for the design of robots which are to perform in a highly dynamic environment.

4 BACKGROUND

This section gives an overview of the technologies which are needed to design and apply the AIMC approach. Intelligent motion and manipulation are also subfields of AI in a complex robotic system. The major aspect of the design of an intelligent robot is the decision making, which takes part as the decision process in the robot loop control. It processes the observed data and can use planning or learning to perform intelligent actions. (Goertzel and Yu, 2014) We describe the state of the art, definitions of level controllers and the principles of reinforcement learning.

https://doi.org/10.1017/dsi.2019.363 Published online by Cambridge University Press

4.1 Motion control

There are different architectures of level controllers which are categorized in bottom, middle and top layers. Low-level controllers are in the bottom layer, are mostly reactive and represent the electromagnetic behavior of robotic hardware components. As a driver, the low-level controller can also act as the trajectory or collision free planner. Mid-level controllers switch between low-level controllers and execute the drivers. The behavior of mid-level controllers can be formulated by Markov Chains. They represent a sequence of possible events in which the probability of each event occurring depends solely on the state attained in the previous event. Markov Chains can be differentiated in Discrete Time Markov Chains (DTMC) and Continuous Time Markov Chains (CTMC). For robotic motion controllers, a Finite State Machines (FSM) is used as a basic mathematical model of computation, which consists of a set of states, transitions and events. (Ridge *et al*., 2017) In a Hierarchical Finite State Machine a state can contain generalized transitions and one or more sub states (Marzinotto *et al*., 2014). It is possible to separate the tasks into sub-tasks, which increases the modularity. Behavior Trees (BT) are a recent alternative to FSM, for creating modular task switching in robot control architectures.(Colledanchise *et al*., 2014) High-level controllers are the highest level of abstraction, and are responsible of executing the planning and monitoring of tasks like scheduler, task planner and Artificial Neural Networks (ANN) (Marzinotto *et al*., 2014). ANN as high-level controllers give robots the ability to switch between states and actions of the mid-level controller.

4.2 Reinforcement learning

Artificial intelligence is the science and engineering of making intelligent machines, consists of different subfields, such as Machine Learning (ML). ML is a field of computer science that gives systems the ability to learn without being explicitly programmed. It consists of the three major fields: unsupervised learning, supervised learning and reinforcement learning(Russel and Norvig, 2010). In reinforcement learning, an agent learns behavior through trial-and-error interactions with a dynamic environment. A Markov Decision Process (MDP) is a mathematical framework to model decision making, instead of describing the solution of a problem. In MDP an agent interacts with its environment and tries to get a reward. It discovers different states by performing certain actions. The goal is to find the best policy that matches to each state and action to get the biggest reward, this total reward can be defined as the sum of the rewards (Kober and Peters, 2012). Three current successful methods like Deep-Q-Learning (DQN) combine Deep Neral Networks with Reinforcement learning (Arulkumaran *et al*., 2017).

4.3 Simulation

Reinforcement learning approaches were historically deployed for video games (Hester *et al*., 2017), but recently got significantly more relevant for simulations (Gu *et al*., 2017). There are several simulation tools which can be used to design an environment. The non-profit organization OpenAI developed an OpenSource library called OpenAI Gym, to bridge reinforcement learning with applications. OpenAI Gym uses a physics simulator for robotic use cases, such as MuJuCo (Plappert *et al*., 2018) which is an advanced physics engine, which allows to build a model that behaves in a highly realistic manner (Brockman *et al*., 2016). V-Rep is a robot simulation tool which has a lower reality physics engine, but a Robot Operating System (ROS) interface (Quigley *et al*., 2009). The well supported ROS interface is provided by Gazebo thanks to the large community. Erlerobotics provides various simulation models (Yan *et al*., 2017) and has built an extended version of OpenAI Gym to combine it with Gazebo (Zamora *et al*., 2016).

5 AI MOTION CONTROL

The traditional way of design is to generate control policies for robotic manipulation tasks using: hard-coded waypoints at the low-level; non-generic states at the mid-level; specialized tasks at the high-level (Figure 2). This approach is repetitive, not scalable and mostly suited for specific applications. Other ways of generating control policies are presented in the section 2. In an industrial context, objects like production parts can have a variety of hundreds or thousands of different items. Applications for these use cases, such as robots with pick and place abilities, require flexibility and high scalability. Instead of developing multiple different policies, the proposed development process of motion control with artificial intelligence can improve the system development lifecycle.

https://doi.org/10.1017/dsi.2019.363 Published online by Cambridge University Press

Systems in a highly dynamic environment need many development iterations; AIMC offers the potential to shorten this development. Control policies can also be improved by using unsupervised or supervised learning, however in this paper we present an approach with reinforcement learning because it gives the robot the ability to learn during its application. The approach requires the design of generic level controllers. Low-level trajectories can be learned by predefined movements, a state machine can consist of given actions and states, moreover, subtasks can be preset in a behavior tree or a task planner. Each motion control layer is connected with a neural network which will be trained hierarchically.

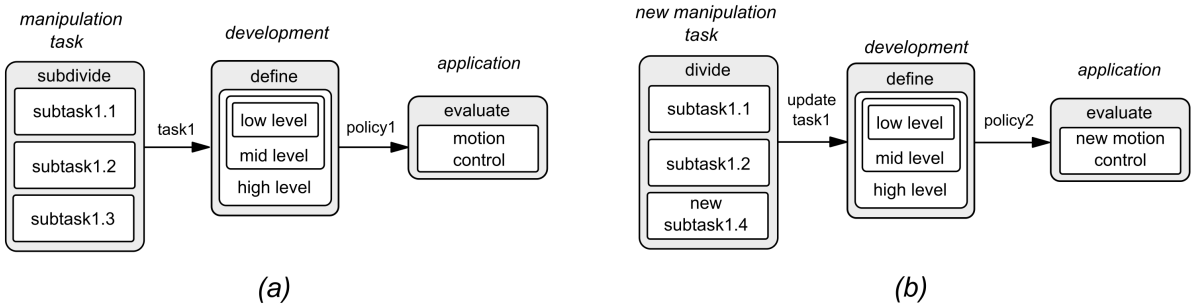

Figure 2. Development of motion control without AI. (a) First and (b) second iteration.

In the beginning of the development, the application will be trained in simulation (Figure 3a). In comparison with manual development, AIMC requires more time in the beginning of the development, but it can save considerable time in following iterations.

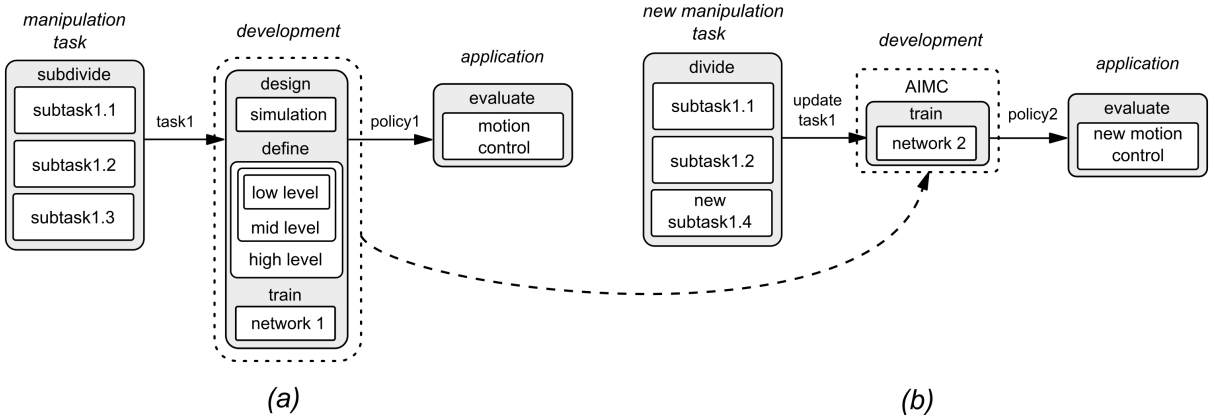

Figure 3. Development of AI-based motion control. (a) First and (b) second iteration.

In a second manipulation task with a new subtask or a different order of subtasks, AIMC can be used again, to train the neural network and to regenerate the motion control policy (Figure 3b). The neural network is connected to the low, mid and high level controllers, and can be changed through the modification of the reward function which represents the manipulation task.

5.1 Design of the approach

Artificial Intelligence based Motion Control (AIMC) uses an open-source toolchain and describes the development of control policies for robotic manipulation tasks. The approach deals with the design of a simulation, the definition of motion controllers and the training and evaluation of the motion control policy. The manipulation task will be divided into subtasks. The motion control can be split into top, mid and bottom layers. The RL method receives these inputs to train actions based on the observations. The sim-to-real bridges ensure the transfer of the policy to the real application (Figure 4). We use the Open Source Tool Gazebo as simulator, but the concept of the approach also works with the other simulation tools. Depending on the use case, the simulation environment can be modified through the implementation of obstacles and handling objects. The Robot Operating System gives the simulation the ability to move and sense. There exists different plugins to implement sensors that measure contact or distance, or to implement a camera that generates a RGB map, a disparity map or a point cloud. There

https://doi.org/10.1017/dsi.2019.363 Published online by Cambridge University Press

are two common different ways of low-level controllers. Joints can be controlled with a PID controller or with the use of a planner. For linear, arm or hybrid robots, there exists linear or rotational motor drives as plugins. In the development of a training script, the implementation of a learning algorithm is needed at first. We use the algorithm of OpenAI Gym. The learning algorithm's reward function describes how the goal can be achieved.

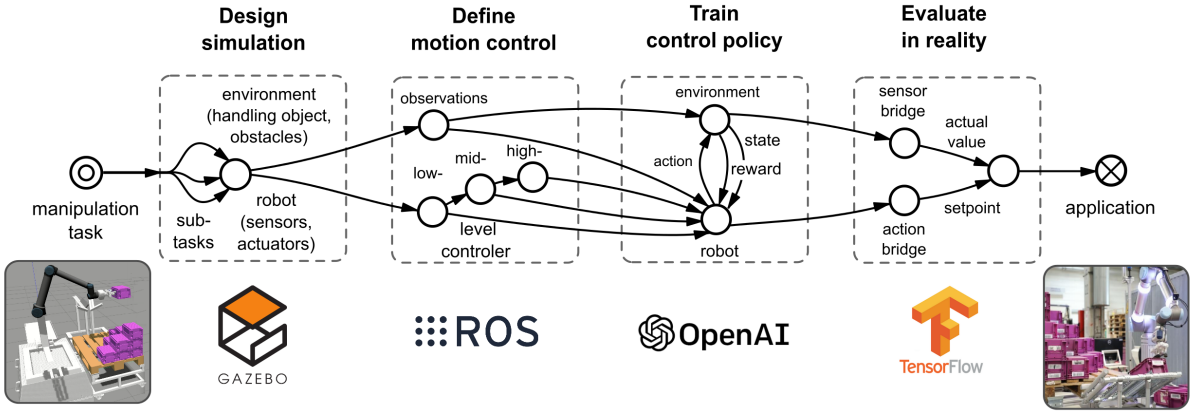

Figure 4. AI-based motion control in four development process steps.

The training of the policy depends on which manipulation task is to be achieved. Basically, the research categorizes the tasks as either reaching, picking, sliding or pushing tasks. The policy can be also extended with different other tasks like placing, stacking or manipulation, like changing positions and rotations of the object itself. Certainly the actions and observations of the simulation vary from the real environment. As such, the goal of transferring the control policy from the simulation to real application is to close the sim-to-real gap. There are different approaches which already overcome this problem, like the dynamic randomization (Peng *et al.*, 2017). There are various use cases which can be used to train certain tasks like the palletizing, depalletizing or packing of objects (Figure 5). These processes are stationary, but the manipulation task can also be applied on mobile robotic systems. Instead of the development of motion control policies for a palletizing robot, the generic approach is designed to train different robotic application and also multiple tasks.

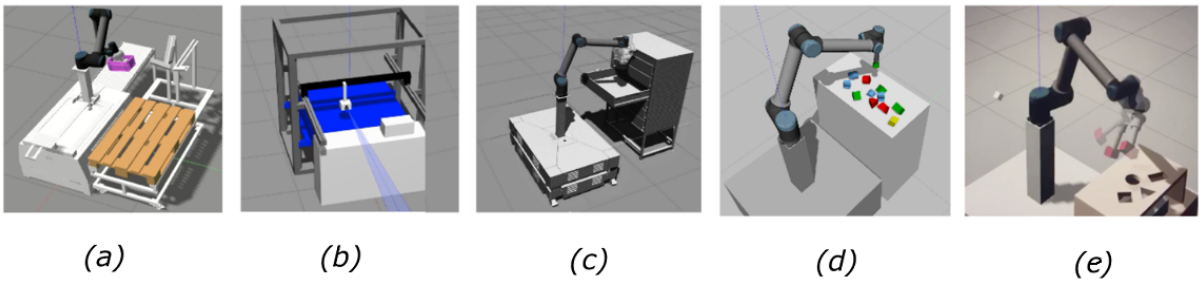

Figure 5. Simulation environments of different applications for real use cases like (a) the palletizing of boxes, (b) the depalletizing of boxes and (c) packing of parts. There are environments like the (d) picking and (e) placing of certain shapes to prove different learning concepts.

6 EXPERIMENTAL WORK

In this research we use one of the simulation and present the application of AIMC to develop low-level motion trajectories for a grasping task. To guarantee the reproducibility of the experiment, we use different object shapes without an industrial context (Figure 5d). The methods developed for reaching, grasping, manipulating or placing an item can be used for all systems modular. We compare the generated policy in simulation with the in reality developed policy. The transfer is beyond the generation of control policies and does not require additional time during the application of the policy therefore we do not consider the transfer of generated policies on the real robotic system in this experiment.

https://doi.org/10.1017/dsi.2019.363 Published online by Cambridge University Press

6.1 Setup

We use a pick-and-place robot that learns to pick various shapes. The setup consists of the robotic arm UR10 from Universal Robots with a vacuum gripper. The robot is controlled by joint coordinates and always has in the beginning of the experiment the same starting pose. The objects are randomly spawned on an area of 80 cm x 40 cm. Using cubes of varying shapes with the size of about 0.5 cm^3. In this experiment, we avoid the use of computer vision for object detection. The simulated robot receives information about the position and a number of the certain item. Potential gripping points of the of items change with their shape.

(a) (b) (c)

Figure 6. The three training samples and seven handling objects (a), the experimental setup in reality (b) and in simulation (c).

The simulation contains the same components as in reality: A model of the robotic arm, the gripper, a table and the objects to be handled. The gripper consists of two contact sensors which simulate the behavior of a vacuum gripper. If both sensors sense contact, this means the object is picked correctly and vacuum could be generated. The robot has six joint movements as actions. The states are: Position of the object and an object number, which both should represent information from a camera system and the joint positions. We use the reinforcement learning approach Deep Q-Learning to train the model. In the following we describe the manual development process and the development of an AI-based process.

6.2 Experimental procedure of the manual development process

The goal is to generate waypoints for a grasping trajectory. The trajectories will be programmed by operators with an expertise in robotics engineering. The operators are training the procedure with three objects and apply their generated experience to develop control policies for the further seven objects. Every new object requires an adaptation of the gripping point. The operators teach the pick-and-place robot in reality by moving the EOAT (end-of-arm tooling) through waypoints. The development time will be represented as working time and can be measured with an MTM (Methods time measurement) analysis (Table 1).

Table 1. MTM analysis for the manual teaching of positions. There are different categories of subtasks.

no.	decomposed movements	task	time (s)	time (TMU)
1	Reaching robot	R	1.3	0.047
2	Graping robot	G	0.4	0.014
3	Move robot in object direction	M	4.7	0.169
4	Focus object postion	ET	0.5	0.018
5	Refine position	P	1	0.036
6	Apply pressure	AP	0.8	0.029
7	Release robot	RL	0.7	0.025
8	Disengage robot	D	0.4	0.014
9	Validate picking	ET	1.9	0.068
		total	11.7	0.42

https://doi.org/10.1017/dsi.2019.363 Published online by Cambridge University Press

We divide the manual operation in the MTM categories: Reach to an object in other hand or to an object in fixed location or on which other hand rests, move (M) the object to an exact location, position (P) with light pressure required and difficult to move and eye focus (ET) time is the time needed to focus the eyes on an object and look at it long enough to verify e.g. the position of the object. The MTM tables and the Time measurement unit (TMU). After training with three sample parts the five experimentees develop for each of the seven parts, which are randomly chosen and placed, the control policy. The experimentees are free to chose the trajectory and it will be saved automatically. The manual handling can be also developed in a simulation environment, but the physical teaching is preferred in this experiment because the operator can create trajectories precisely.

6.3 Experimental procedure of the AI-based development process

The designed toolchain provides a program to train and to use the motion control policy. The first procedure is to train the robot to pick three objects. The objects' shape and position will be spawned at random. During the training the robot learn to pick the same three objects like in the manual experiment. The robot trains for 1,500 episodes to pick these objects. After it has a success average of over 80% the model will be saved and stopped. In the application and experimentation program, the trained model will be applied for the other seven objects. The time between spawning the objects and successful reaching it, will be measured.

7 RESULTS

The results of the experiments are shown in figure 7. The simulated robot is able to learn trajectories to pick certain objects. After 1,000 episodes the robot reached a reward average of 180 per 100 episodes (Figure 7a) and a success average of 82% (Figure 7b). An human operator has an average working time of 11.7 s as shown in table 1. However, there are differences between the development time of various items. The learned model was able to generate a successful trajectory after 1.82 s.

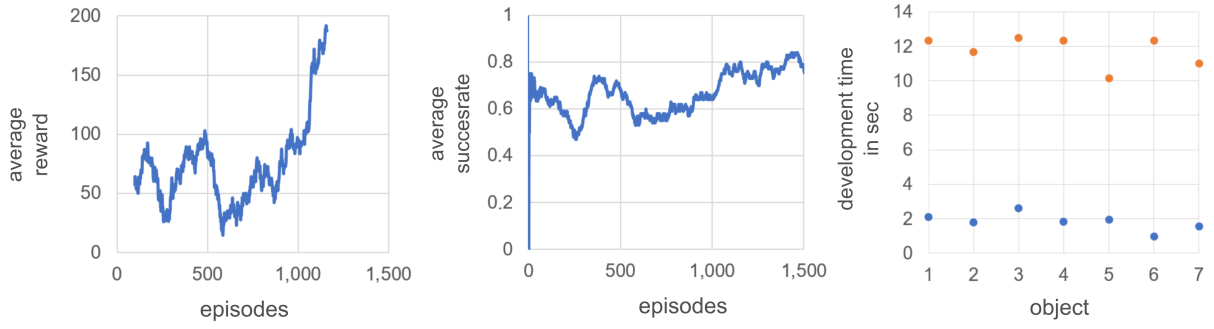

Figure 7. Average reward (a), average success rate (b) and the time for the development of low-level trajectories generated by human in orange and simulated robot in blue (c).

8 CONCLUSION AND FUTURE WORK

This paper has shown the current state of the art in AI-based motion controllers. We reveal the gap in the development of robotic motion control policies with manipulation tasks in highly dynamic environments and present AIMC, as a solution for the development, which, on the one hand, gives developers the possibility to generate dynamic motion controllers, and on the other hand, gives a robot the ability to update motion controllers on its own. A recent alternative is the manual implementation and modification of motion controllers, which causes a time-consuming development. This paper has given an introduction to the identification, but has also shown the potential of descreasing the development time. The factors of the descriptive study I lead to the result of a reference model, here described as the AIMC approach. This theory will be proven in the prescriptive study, which has an impact model as output and the systematic development with AIMC will be dealt with in further publications.

https://doi.org/10.1017/dsi.2019.363 Published online by Cambridge University Press

REFERENCES

Arulkumaran, K., Deisenroth, M.P, Brundage, M. and Bharath, A.A. (2017), "Deep reinforcement learning: A brief survey", *IEEE Signal Processing Magazine*, Vol. 34 No. 6, pp. 26–38. https://doi.org/10.1109/msp.2017.2743240.

Bai, T., Yang, J., Chen, J., Guo, X., Huang, X. and Yao, Y. (2017), "Double-task deep q-learning with multiple views", *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 1050–1058. https://doi.org/10.1109/iccvw.2017.128.

Beer, M. (2018), "Artificial Intelligence, Robotics and 'Autonomous' Systems", *European Group on Ethics in Science and New Technologies*. https://doi.org/10.2777/786515.

Blessing L. and Chakrabarti A. (2009), In *"Design Research Methodology", Springer-Verlag London*, p. 39. https://doi.org/10.1007/978-1-84882-587-1.

Bousmalis, K., Irpan, A., Wohlhart, P., Bai, Y., Kelcey, M., Kalakrishnan, M., Downs, L., Ibarz, J., Pastor, P., Konolige, K., Levine, S. and Vanhoucke, V. (2017), "Using simulation and domain adaptation to improve efficiency of deep robotic grasping", *CoRR*, abs/1709.07857. https://doi.org/10.1109/icra.2018.8460875.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J. and Zaremba, W. (2016), "Openai gym", *CoRR*, abs/1606.01540.

Colledanchise, M., Marzinotto, A., and Ögren, P. (2014), "Performance analysis of stochastic behavior trees", *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3265–3272. https://doi.org/10.1109/icra.2014.6907328.

Clavera, I., Held, D., and Abbeel, P. (2017), "Policy transfer via modularity and reward guiding", *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1537–1544. https://doi.org/10.1109/iros.2017.8205959.

Ertel, W., Schneider, M., Cubek, R., and Tokic, M. (2009), "The teaching-box: A universal robot learning framework", *2009 International Conference on Advanced Robotics*, pp. 1–6.

European Group on Ethics in Science and New Technologies (2014), "Toward a Framework for Levels of Robot Autonomy in Human-Robot Interaction", *Journal of Human-Robot Interaction*, Vol. 3 No. 2. https://doi.org/10.5898/jhri.3.2.beer.

Ghadirzadeh, A., Maki, A., Kragic, D. and Björkman, M. (2017), "Deep predictive policy training using reinforcement learning", *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2351–2358. https://doi.org/10.1109/iros.2017.8206046.

Goertzel, B. and Yu, G. (2014), "From here to agi: A roadmap to the realization of human-level artificial general intelligence", *2014 International Joint Conference on Neural Networks (IJCNN)*, pp. 1525–1533. https://doi.org/10.1109/ijcnn.2014.6889801.

Gu, S., Holly, E., Lillicrap, T. and Levine, S. (2017), "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates", *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3389–3396. https://doi.org/10.1109/icra.2017.7989385.

Guerin, K.R., Lea, C., Paxton, C., and Hager, G.D. (2015), "A framework for end-user instruction of a robot assistant for manufacturing", *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6167–6174. https://doi.org/10.1109/icra.2015.7140065.

Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Sendonaris, A., Dulac-Arnold, G., Osband, I., Agapiou, J., Leibo, J.Z., and Gruslys, A. (2017), "Learning from demonstrations for real world reinforcement learning", *CoRR*, abs/1704.03732.

Hersch, M., Guenter, F., Calinon, S. and Billard, A. (2008), "Dynamical system modulation for robot learning via kinesthetic demonstrations", *IEEE Transactions on Robotics*, Vol. 24 No. 6, pp. 1463–1467. https://doi.org/10.1109/tro.2008.2006703.

International Federation of Robotics. (2018), "Executive Summary World Robotics 2018 Industrial Robots". *World Robotics 2018*, pp. 13–22.

Johannsmeier, L. and Haddadin, S. (2017), "A hierarchical human-robot interaction-planning framework for task allocation in collaborative industrial assembly processes", *IEEE Robotics and Automation Letters*, Vol. 2 No. 1, pp. 41–48. https://doi.org/10.1109/lra.2016.2535907.

Kemp, C.C, Edsinger, A. and Torres-Jara, E. (2007), "Challenges for robot manipulation in human environments [grand challenges of robotics]", *IEEE Robotics Automation Magazine*, Vol. 14 No. 1, pp. 20–29. https://doi.org/10.1109/mra.2007.339604.

Kober, J. and Peters, J. (2012), *"Reinforcement Learning in Robotics: A Survey"*, pp. 579–610. Springer Berlin Heidelberg, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-27645-3_18.

Kok, Z.K.J., Causo, A., Chong, Z.H. and Chen, I.M. (2017), "Designing modular robotic architecture for e-commerce bin picking task fulfillment", *TENCON 2017 - 2017 IEEE Region 10 Conference*, pp. 1109–1114. https://doi.org/10.1109/tencon.2017.8228023.

https://doi.org/10.1017/dsi.2019.363 Published online by Cambridge University Press

Levine, S., Finn, C., Darrell, T. and Abbeel, P. (2015), "End-to-end training of deep visuomotor policies", *CoRR*, abs/1504.00702.
Levine, S., Wagener, N. and Abbeel, P. (2015), "Learning contact-rich manipulation skills with guided policy search", *CoRR*, abs/1501.05611. https://doi.org/10.1109/icra.2015.7138994.
Li, Q., Haschke, R. and Ritter, H. (2015), "A visuo-tactile control framework for manipulation and exploration of unknown objects", *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 610–615. https://doi.org/10.1109/humanoids.2015.7363434.
Mar, T., Tikhanoff, V., Metta, G. and Natale, L. (2015), "Multi-model approach based on 3d functional features for tool affordance learning in robotics", In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 482–489. https://doi.org/10.1109/humanoids.2015.7363593.
Marzinotto, A., Colledanchise, M., Smith, C. and Ögren, P. (2014), "Towards a unified behavior trees framework for robot control", *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5420–5427. https://doi.org/10.1109/icra.2014.6907656.
Nguyen-Tuong, D. and Peters, J. (2011), "Model learning for robot control: a survey", *Cognitive processing*, Vol. 12, pp. 319–340. https://doi.org/10.1007/s10339-011-0404-1.
Peng, X.B., Andrychowicz, M., Zaremba, W. and Abbeel P. (2017), "Sim-to-real transfer of robotic control with dynamics randomization", *CoRR*, abs/1710.06537. https://doi.org/10.1109/icra.2018.8460528.
Pfeiffer, S. (2016), "Robots, industry 4.0 and humans, or why assembly work is more than routine work", *Societies*, Vol. 6 No. 2. https://doi.org/10.3390/soc6020016.
Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J.T, Chociej, M., Welinder, P., Kumar, V. and Zaremba, W. (2018), "Multi-goal reinforcement learning: Challenging robotics environments and request for research", *CoRR*, abs/1802.09464.
Quigley, M., Gerkey, B.P., Conley, K., Faust J., Foote, T., Leibs, J., Berger, E., Wheeler, R. and Andrew Ng. (2009), "ROS: an open-source robot operating system".
Ridge, B., Gaspar, T. and Ude, A. (2017), "Rapid state machine assembly for modular robot control using meta-scripting, templating and code generation", *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 661–668. https://doi.org/10.1109/humanoids.2017.8246943.
Russell, S. and Norvig, P. *"Artificial Intelligence - A Modern Approach"*, Pearson Higher Education, New Jersey.
Rusu, A.A, Vecerik, M., Rothörl, T., Heess, N., Pascanu, R. and Hadsell, R. (2016), "Sim-to-real robot learning from pixels with progressive nets", *CoRR*, abs/1610.04286.
Sasaki, K., Noda, K. and Ogata, T. (2016), "Visual motor integration of robot's drawing behavior using recurrent neural network", *Robotics and Autonomous Systems*, Vol. 86, pp. 184–195. https://doi.org/10.1016/j.robot.2016.08.022.
Tai, L. and Liu, M. (2016), "A robot exploration strategy based on q-learning network", *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pp. 57–62. https://doi.org/10.1109/rcar.2016.7784001.
Torrado, R.R, Bontrager, P. and Perez-Liebana, D. (2018), "Deep reinforcement learning for general video game ai", *CoRR*, abs/1806.02448. https://doi.org/10.1109/cig.2018.8490422.
Wisskirchen, G. IBA Global Employment Institute. (2017), "Artificial intelligence and robotics and their impact on the workplace".
Yan, Z., Fabresse, L., Laval, J. and Bouraqadi, N. (2017), "Building a ros-based testbed for realistic multi-robot simulation: Taking the exploration as an example", *Robotics*, Vol. 6 No. 3. https://doi.org/10.3390/robotics6030021.
Zadeh, S.M.K. (2012), "A dynamical system-based approach to modeling stable robot control policies via imitation learning".
Zamora, I., Lopez, N.G., Vilches, V.M. and Cordero, A.H. (2016), "Extending the openai gym for robotics: a toolkit for reinforcement learning using ROS and gazebo". *CoRR*, abs/1608.05742.

https://doi.org/10.1017/dsi.2019.363 Published online by Cambridge University Press