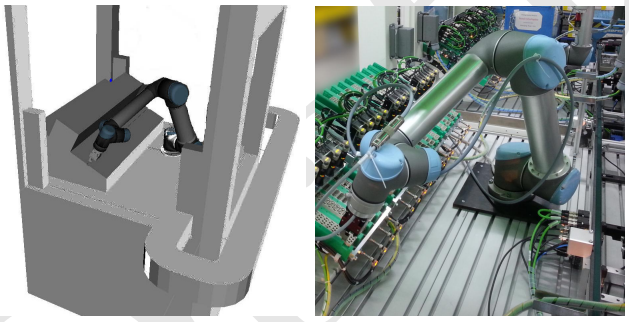
# Adaptive Handling Assistance for Industrial Lightweight Robots in Simulation

Agathe Balayn, Jeffrey Frederic Queißer, Michael Wojtynek, Sebastian Wrede
Research Institute for Cognition and Robotics (CoR-Lab) & Faculty of Technology
Bielefeld University, Universitätsstr. 25, 33615 Bielefeld, Germany
Email: {abalayn, jqueisse, mwojtynek, swrede}@cor-lab.uni-bielefeld.de, www.cor-lab.de

***Abstract*— With the growth of lightweight robots in industry handling assistance becomes more and more important for solving industrial tasks. We present an adaptive compliance control mode for industrial robots based on a learned equilibrium model. This enables us to cope with changing manufacturing environments, attached grippers as well as devices with inaccurate dynamic models, e.g. stiff tubes, wires, protection shields or hose packages. A further feature of the proposed method is the expandability by an additional parameterization that allows to deal with task variability. In this work we evaluate our approach using the example of a task that incorporates variable payloads. All experiments are conducted in a simulation framework to evaluate the feasibility of the proposed approach for industrial robot scenarios.**

## I. INTRODUCTION

A flexible production that targets small lot sizes requires flexible usage of industrial lightweight robots. Lightweight robots [1] are used in production systems to close automation gaps where a full automation is not profitable, reasonable or possible. Further, they are supposed to support humans in manufacturing tasks. Therefore, robots and their control systems offer modes to enable intuitive interaction with the human worker. Common modes for interaction are for example gravity compensation and joint impedance mode. These modes are able to react to external forces and compensate the effect of gravity on the robot. For instance, the human worker operates in gravity compensation to perform pick and place tasks of heavy objects with assistance of the robot. In that case, gravity compensation balances the weight of the robot and the load with a counter force to make it moveable with less effort. However, gravity compensation requires exact knowledge about the weight of the robot and work loads. Otherwise, motions of the robot become unpredictable and dangerous for the human worker and the payload. To deal with this problem, the weight of the tool as well as the manipulated object have to be declared to the robot system in advance. This additional effort is a drawback for flexible robot systems that have to adapt quickly to frequently changing tasks in modern manufacturing environments. Although methods for model estimation and parameter search for models of rigid body dynamics exist, e.g. [2]–[4], they cannot be applied in all cases. For example, high flexibility in the workplace design of the robot leads to an individual configuration setup: E.g. stiff cables, soft grippers or protective shields attached to the actuators. Without an appropriate model of those structures, uncertainties in the dynamics of the robot can occur. A proper application of lightweight robots is not feasible in this situation. Our proposed approach is able to handle unknown weights for an adaptive assistance mode in a constrained workspace. To tackle this issue, we transfer the methods developed in [5] for a soft and continuous robot to a lightweight robot and its corresponding industrial scenario. Therefore, we estimate an equilibrium model of the robot that is able to cope with uncertainties caused by cables, protective shields or welding hose packages which are attached to the robot body. An additional contribution of this work is the extension of the control methods, presented in [5], to a parameterized configuration of the robot, i.e. variable payloads at the end-effector. For evaluation, we present a simulation driven approach incorporating a real-time physics simulation. The implementation and application of the adaptive handling system splits into two stages. In the first stage we sample the data from random robot movements in the constrained workspace including joint angles and torques. The data is sampled with representative and known payloads. In a second stage an interaction component reacts to external and unknown forces by comparing the predicted and real torques for the current joint configuration.



(a) (b)

*Fig. 1: Simulation setup (a) and constrained application area (b) of the UR5 industrial lightweight robot. High flexibility in the workplace design of the robot as well as individual configuration leads, e.g. cables, protective shields leads to uncertainty in robot model.*

The remaining paper is structured as follows. Section II presents related work to this approach. Our learning methods, control composition as well as software architecture for sim-

ulation are described in Section III. Section IV shows details of the experimental setup and the acquisition of data sets. The performance of the experiment with payload variations and the according results are described and discussed in Section V and summarized in Section VI.

## II. Related Work

Modularized production cells are a field of application for adaptive handling assistance where the workspace is limited, rearranged and reconfigured with frequently changing demands of small batch products [6]. The lightweight robot manipulator in this approach is used as a versatile handling subsystem for a customer specific production down to a lot size of one. Regarding dynamic model estimation for gravity compensation, [7] shows the implementation of an adaptive PD controller which results in stability with limited knowledge about the system. It relies on the lower and upper bounds of the inertia matrix, while the inertia and gravity parameters or the payload are unknown, and the PD gains are less precisely tuned. [8] presents an adaptive controller for unknown inertia parameters which is globally convergent. Tests are conducted in simulation on a 2-DOF robot manipulator. However, our study is based on a 6-DOF robot, which introduces more complexity for the robot control. [9] and [10] propose a neural network adaptive control which is able to work with variations within the robot configuration and with unknown payloads, as proved in [11] with experiments on a PR2 robot. Although this method handles unknown weights efficiently in trajectory tracking mode, it does not implement an adaptive handling control mode. [12] and [13] introduce an approach that handles unknown weights which are not solved by either model-based compensation or PID control. According to the authors, a PID control needs fine tuning of gains in order to achieve good performance in the workspace. The authors propose a method for generating exact gravity compensation at a desired set-point, without the knowledge of the robot's dynamic model. However, simulations are presented for a 3R rigid robot arm moving in a vertical plane that is different from our approach on a 6-DOF robot in three dimensional space. [14] shows a payload identification approach for online programming of industrial robots. This method is derived from a robot identification method that allows to integrate the experiment design with geometry and inertia simulation, signal processing, and parameter estimation. Model estimates that target higher DOF robot can be found for example in [2]–[4], but they focus on parameter estimation for inverse models as well and cannot deal with changing and not existing dynamic models. For the realization of an interactive control without the knowledge of a dynamic model, a hybrid approach was introduced in [5] that combines classical control modes, like PID control, and learning elements. In contrast to the approach of our contribution, it is demonstrated how an inverse equilibrium model can be learned and effectively exploited for quick and agile interactive control on a continuum soft robot. To gain a powerful control architecture for this soft continuum robot, a simulation, analytic tools and middleware have been integrated [15]. Indeed, this approach shows that soft robots as well as non-soft robots, like the UR5, are globally asymptotically stable with different controllers, however it requires deep knowledge on the dynamics parameters of the system and precise tuning.

## III. Model Learning and Control Architecture

This work refers to previous experiments focusing on soft robots [5], showing an implementation of a gravity-compensation-like mode for soft robots whose dynamic equations are unknown. Therefore, we consider a simplified model, which is restricted to the mechanical equilibrium points of the robot's dynamics. The target platform for experiments, Universal Robot UR5, has six rotational joints, thus we consider an equilibrium model with six joint angles as input and six joint torques as output.
Further, we utilize a simulation setup, described in Sec. III-C, to evaluate the feasibility of the control scheme for this industrial lightweight robot. In comparison to the soft robot, equilibrium points are not achieved by applying constant torques $\boldsymbol{\tau}^*$ until convergence of the joint angles. We utilize an integrated PID controller and sample joint torques of random postures in the workspace. In such a state, neither angles nor torques at each joint change over time: $\dot{\boldsymbol{\tau}} = \dot{\mathbf{q}} = \ddot{\mathbf{q}} = 0$.

$$\boldsymbol{\tau}^* = \mathbf{f}(\mathbf{q}^*, 0, 0) \Leftrightarrow \hat{\boldsymbol{\tau}}(\mathbf{q}^*) = \boldsymbol{\tau}^* \ , \tag{1}$$

where $\hat{\boldsymbol{\tau}}$ denotes the inverse equilibrium model that represents the direct relation between joint angles $\mathbf{q}^*$ and torques $\boldsymbol{\tau}^*$. In case that a high deflection of the model estimate and the currently sensed torques is observed: $|\hat{\boldsymbol{\tau}}(\mathbf{q}^*)_i - \boldsymbol{\tau}_i^*| > T$, i.e. they exceed threshold $T$, the target angle $\mathbf{q}_i$ of the $i$-th joint gets updated. The updated joint angles are estimated by $\Delta\mathbf{q}_i = \boldsymbol{\beta}_i(\boldsymbol{\tau}(\mathbf{q}^*)_i - \boldsymbol{\tau}_i^*)$ with $\beta$, a scalar factor transforming prediction error to joint angle update. Therefore the joint configuration of the robots follows external forces that cause an unexpected joint torque.

### A. Model Learning

For the inverse equilibrium model, the Extreme Learning Machine (ELM) [16] comprises $\mathbf{q} \in \mathbb{R}^{I=6}$ input, $\mathbf{h} \in \mathbb{R}^R$ hidden, and $\hat{\boldsymbol{\tau}} \in \mathbb{R}^{O=6}$ output neurons. The input is connected to the hidden layer by the input matrix $W^{\mathbf{inp}} \in \mathbb{R}^{R \times I}$. The read-out matrix is given by $W^{\mathbf{out}} \in \mathbb{R}^{O \times R}$. For input $\mathbf{q}$, the output of neuron $o$ is computed by

$$\hat{\boldsymbol{\tau}}_o(\mathbf{q}) = \sum_{j=1}^{R} W_{oj}^{\mathbf{out}} f(\sum_{k=1}^{I} W_{jk}^{\mathbf{inp}} \mathbf{q}_k + b_j) \ , \tag{2}$$

where $b_j$ is the bias for neuron $j$, and $f(x) = (1 + e^{-x})^{-1}$ the logistic activation function. The components of the input matrix $W^{\mathbf{inp}}$ and the biases $b_j$ are drawn from a random distribution and remain fixed after initialization.

Let $\mathcal{D} = (A, T) = (\alpha^k, \tau^k)$ with $k = 1 \dots N_{\text{tr}}$ be the data set for training, where $N_{\text{tr}}$ is the number of training samples. $A \in \mathbb{R}^{I \times N_{\text{tr}}}$ is the collection of angles, and $T \in \mathbb{R}^{O \times N_{\text{tr}}}$ is the matrix of target torques for all $N_{\text{tr}}$ samples.

Supervised learning is restricted to the read-out weights $W^{\mathbf{out}}$ and accomplished by solving ridge regression [17]:

$$\mathbf{W^{out}} = argmin_{\mathbf{W}}(\|\mathbf{H(A)W^T} - \mathbf{T}\|^2 + \lambda\|\mathbf{W}\|^2) \ , \quad (3)$$

where $H(A) \in \mathbb{R}^{R \times N_{\text{tr}}}$ is the matrix collecting the hidden-layer states. The growth of the read-out weights is controlled by the regularization parameter $\lambda$.

*Parameterized Model Learning:* Additionally to the estimation of an equilibrium model for fixed load configurations (III-A), we investigate a parameterization of the learned equilibrium model. Such a parameterization is able to cover variability in the robot or task configuration. We compare two approaches for learning of parameterized models that we explain in the following:

*a) Extended Input-Space:* The first method is based on the idea of extending the input space by adding payload as an additional input. The output remains to be mapped towards the six joint torques. In order to evaluate payload prediction, we also evaluated an ELM learner having joint angles and joint torques as inputs and payload as predicted output.

*b) Regression in the Model Space:* After having learned the inverse equilibrium models for different payloads, we learn with the second method a mapping from payload parameters to model parameters, as shown in Fig. 2 [18]. We consider the model parameters as output weights $W_S^{out}$ of the specialist ELM. The remaining parameters, i.e. input weights and offsets, of each specialist ELM are considered as being equal to each other. A second ELM, the generalist, is used for regression from payload to $W_S^{out}$. The readout weights $W_G^{out}$ are estimated with ridge regression, as previously mentioned in Eq. 3. We expect that this method will require fewer training data as the input dimension is smaller. As a result, the ELM generalist parameterizes the ELM specialist by computing its output weights $\mathbf{W_s^{out}}$ from the input payload $\mathbf{p_i}$. The ELM specialist is responsible for the mapping to output torques corresponding to the input six joint angles.

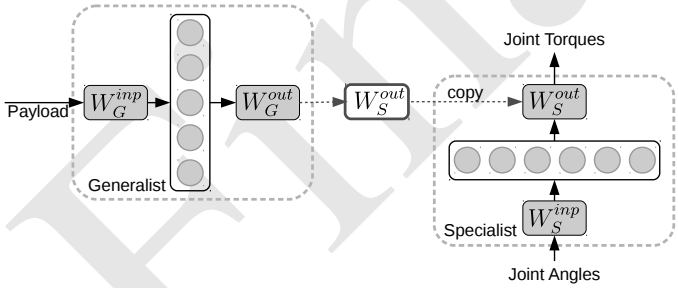



*Fig. 2: Concept of regression in the model space as proposed in [18]. The generalist learner estimates the read-out weights for the specialist learner. Resulting in a generalization in the space of read-out weights.*

## B. Control Architecture

The control architecture for the proposed assistive handling mode is depicted in Fig. 3. The equilibrium model (1) estimates the expected torque $\hat{\boldsymbol{\tau}}$ for the current configuration $\mathbf{q}^*$ of the robot. The interaction module (3) estimates a desired posture update $\mathbf{q}$ based on the error (2) of the predicted $\hat{\boldsymbol{\tau}}$ and real torques $\boldsymbol{\tau}^*$ of the robot. In case an error threshold is exceeded (4) the current target joint angles (5) are updated or kept unchanged. This ensures that no drifts of the robot actuator occur that are caused by noise or inaccuracies of the learned the equilibrium model. The integrated PID position controller (6) updates the real torques $\mathbf{q}$ that are sent to the Simulator (7) and the robot model.

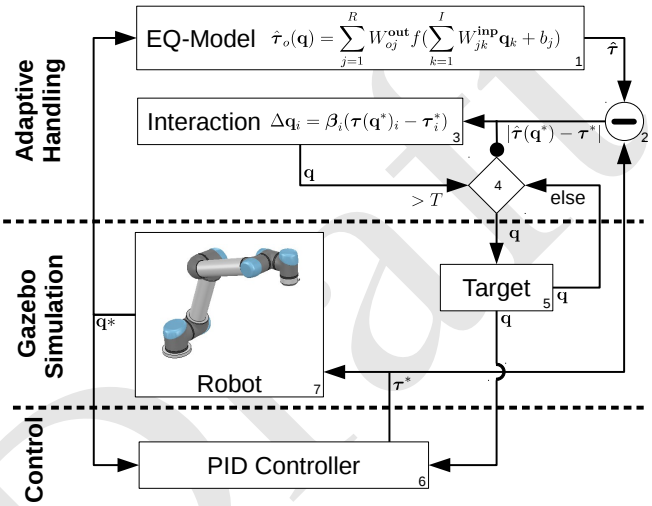



*Fig. 3: Proposed control architecture for equilibrium based assistive handling modes on lightweight robots. Equilibrium model (1), estimation of prediction error (2), posture update based on prediction error (3), threshold based activation of posture update (4), desired joint angles (5), position controller (6) and robot simulator (7).*

## C. Software Architecture

This section describes the software architecture that visualizes, simulates and controls the UR5 robot. For our experiments with the adaptive handling assistance we use the robot simulator Gazebo [19] to simulate the real robot behaviour. In general, the software architecture is implemented within the robotic framework OROCOS (Open Robot Control Software Project) [20] and its Real-Time capable Toolkit (RTT) [21]. RTT allows an inter-component communication with output and input ports that exchange data between the components. In our case the components mostly provide and receive a six dimensional joint vector corresponding to the six rotational joints of the UR5 robot.

Moreover, the software architecture of the UR5 handling assistance represents a component structure that distributes functionalities of the handling framework to each component. Components which are taken into account for handling an unknown weight are a PID controller, Data Collector and an Interaction component. In our setup the PID Controller substitutes the behaviour of the hardware controller and controls the robot by joint torques. Therefore, the input of the PID controller component takes the desired target joint configuration and moves the robot by applying torques on the robot joints.

The Data Collector component is switched on while collecting sample data from the simulated UR5 robot in Gazebo.

During this sampling, the mapping between joint torques and the corresponding joint configuration is recorded. If the robot is trained with a sample weight for later use with the adaptive handling mode, the mass at end-effector is recorded as well. Further, the interaction component provides the handling system with the trained equilibrium model via its port. A steady comparison of joint torques in each joint configuration with the provided torques from the Interaction component allows the adaptive handling system to follow external forces. The software architecture combining robot control with OROCOS and simulation with Gazebo offers the user to test his experimental setup. The presented framework architecture allows to integrate models and machine learning algorithms for physical simulation. Sampling the data is possible with less effort than on a real robot and could be done on several machines simultaneously. The sampled data, e.g. the joint torques, are compared and checked towards plausibility to UR5 joint torques on the real robot. Further, sampling data in simulation avoids safety issues like collisions.

## IV. EXPERIMENTAL SETUP

For the experimental evaluation we record a training data set required for the estimation of the equilibrium model. A distinct test data set is used for evaluation of the quality of the equilibrium model. As argued in [5], the quality of the equilibrium influences the threshold for posture update and therefore interaction quality. The better the model approximation of the equilibrium states of the UR5 robot, the lower the threshold $T$ and consequently the higher is the sensitivity of the robot to external forces. In case $T$ is chosen to low, system noise triggers a position update which results in undesired drifts of the robot. Further evaluations target the interaction quality by analyzing the sensitivity of the robot in the real application, i.e. simulation. We evaluate the generalization capabilities to fixed load configurations in a first step in Section V-A, followed by experiments targeting variable load conditions in V-B. For the variable load configuration we evaluate the quality of torque prediction as well as the prediction of the current load situation.

### *Acquisition of Data Sets*

To ensure data recording in an equilibrium state, the robot's positions and joint torques are measured 3.5 seconds after the joint command is sent in order to reach torque stabilization. The recording constitutes of random positions that are chosen between specific intervals from inside the robot workspace. Overall 15625 random positions are collected, as we collect 5 random joint angles and the robot has 6-DOF ($5^6 = 15625$). Joint positions are joint angles measured in radians and joint torques are measured in Newton meters. We split the collected data set randomized into equally sized training and test sets. As shown in Fig. 1 the real robot has to operate in a constrained workspace.

The given joint angle ranges, shown in Table I, ensure save operation and valid robot configurations in the workspace without self-collision or collision with the environment.

*TABLE I: Workspace ranges for each joint angle $q_i$ with $\Psi = q_2 - acos(sin(-q_2)\frac{l_1}{l_2})$ that prevents the UR5 robot of self-collisions or collisions with the environment.*

| Joint $q_i$ | Minimum Angle [rad] | Maximum Angle [rad] |
|---|---|---|
| 1 | 0 | 6.28 |
| 2 | $-0.1$ | $-2.3$ |
| 3 | $0.87 - \Psi$ | $-0.77 - \Psi$ |
| 4 | $-3.14$ | 0.7 |
| 5 | $-1.4$ | 1.57 |
| 6 | $-1.57$ | 3.14 |

*Variable Payload:* For the evaluation of variable load configurations, we record a new data set containing joint torques and joint positions with respect to the parameterization, i.e. payload at the end-effector. For the evaluation data set, we use the same random sampling as before with an additional randomization of the payload between 0-5 kg. Further we record a data set with payloads between 5-10 kg to evaluate the extrapolation performances of the algorithms beyond the real robot hardware limits. Due to the requirements of the model space learning we cannot use random payload samples for training. Therefore we can refer to the data set of fixed payloads. Such data reflects the conditions expected for the scenario in real application cases, since in a common use case only a set of fixed conditions are available for training. Therefore, our training data set includes payloads of 0, 1, 3, 5 kg over randomized robot configurations as before.

## V. EXPERIMENTS

As described in Section III-B and III-C, evaluation is done for four experiments: First we evaluate the assistive mode for fixed payload conditions. Subsequent experiments target variable payload conditions, therefore we evaluate the predictability of the payload by observing joint angles and torques. Finally we compare two methods for the parameterized estimation of expected joint torques.

### *A. Inverse Equilibrium Model for Fixed Payloads*

First, we evaluate the applicability of learning an inverse equilibrium model of the robot specific to each payload. The learning is evaluated for 0 kg (no payload), 1 kg, 3 kg, and 5 kg payloads attached to the end-effector.

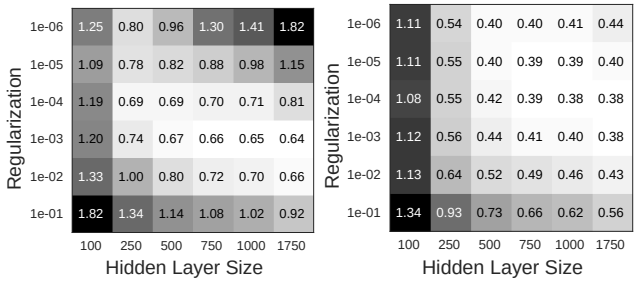
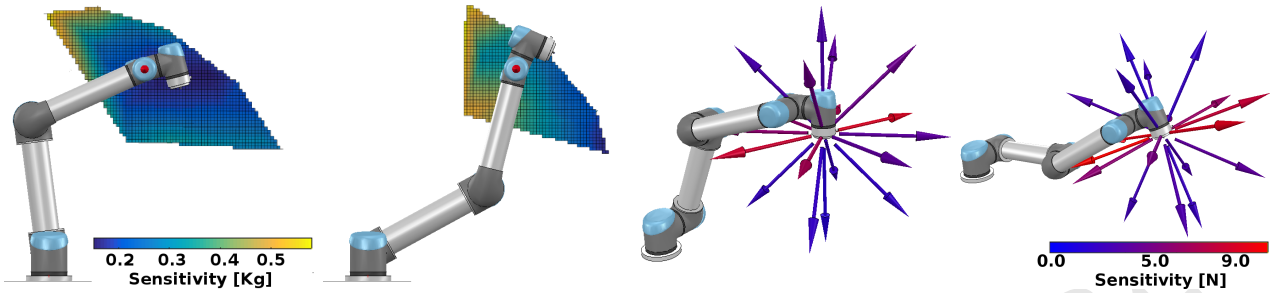



*Fig. 4: Results for parameter optimization with no payload. Given values represent the mean euclidean error of joint 2 to 5 for five-fold cross validation on (a) $N = 1000$ and (b) $N = 7500$ random selected samples of the recorded data set.*




*(a)* *(b)* *(c)* *(d)*

*Fig. 5: Visualization of the sensitivity of posture updates to external forces. We show the sensitivity to loads at the end-effector for postures with positive (a) and negative (b) angles of $q_3$. The sensitivity is shown for posture changes of $q_2$ and $q_3$ inside the workspace. Depending on the posture various sensitivities are achieved. The sensitivity to directed forces for two sample configurations are shown in (c) and (d), due to the redundancy of the robot, different sensitivities for the same end-effector position can be achieved.*

Since joint 1 (robot base) and joint 6 (end-effector) are not affected by gravity, the mean prediction for those joints have to be zero over random postures. Therefore we focus the evaluation on the remaining 4-DOF of the robot.

*Mapping of Joint Positions and Joint Torques:* An ELM learner, as described in Sec. III-A, is utilized in order to learn the equilibrium model. The inputs are the current joint angle configurations of the robot and desired outputs are the associated torques for each joint. To find a sufficient parametrization for model learning, we conducted a grid search with five-fold cross validation. Fig. 4 shows the result for variable hidden layer size and regularization $\lambda$. Performance of the ELM is evaluated in terms of MSE (Mean Squared Error) on the test set for each payload. The results show that a low regularization in combination with a small amount of training samples can lead to a reduced performance, most probably caused by overfitting.

*TABLE II: Torque Error for each Payload and each Joint.*

| | 0 kg | 1 kg | 3 kg | 5 kg |
|---|---|---|---|---|
| Joint 1 MSE | 0.07129 | 0.1405 | 0.2854 | 0.4953 |
| Variance | 0.1426 | 0.2809 | 0.5707 | 0.9905 |
| Joint 2 MSE | 0.06367 | 0.05289 | 0.1734 | 0.7105 |
| Variance | 0.1273 | 0.1058 | 0.3466 | 1.421 |
| Joint 3 MSE | 0.06073 | 0.03215 | 0.0757 | 0.2675 |
| Variance | 0.1214 | 0.06429 | 0.1514 | 0.535 |
| Joint 4 MSE | 0.3849 | 0.1248 | 0.07985 | 0.1605 |
| Variance | 0.7699 | 0.2494 | 0.1597 | 0.321 |
| Joint 5 MSE | 0.002545 | 0.006557 | 0.02558 | 0.005091 |
| Variance | 0.002545 | 0.01311 | 0.05166 | 0.1362 |
| Joint 6 MSE | 0.7739 | 0.7546 | 0.7335 | 0.9231 |
| Variance | 1.548 | 1.509 | 1.467 | 1.846 |

Based on the evaluation of the cross validation results, we have selected the parameterization of the learner. Further, Table III presents the parameter choice for each trained payload. We decided between a compromise of low performance degradation and a small hidden layer size for $N = 7500$ training samples. To evaluate performance of the ELM for each payload, the whole evaluation procedure is repeated ten times and the mean and variance of the MSE is computed. Table II presents the joint MSE and the variance obtained for each payload on the evaluation set trained on $N = 5000$ random samples of the training set. It

*TABLE III: ELM Parameter Optimization.*

| | 0 kg | 1 kg | 3 kg | 5 kg |
|---|---|---|---|---|
| Hidden Dimension | 500 | 500 | 750 | 750 |
| Regularization | $10^{-5}$ | $10^{-5}$ | $10^{-5}$ | $10^{-5}$ |

can be observed that the error and variance increase with the payload, since the torque range increases with the payload at the end-effector as well. However, the error of joint 4 decreases with heavier payloads. In this case the payload force stabilise the behaviour of the PID controller in certain joint configurations.

*Threshold Estimation:* Threshold $T$ (Eq. 1) allows an update of the robot posture in the adaptive handling mode during simulation. Therefore, the end-effector payload is set to the model payload and the robot is brought to ten different fixed positions. The torque difference for each joint is read. For most positions, the torque difference is in the range of errors in Table II. However, for extreme configurations in which the robot arm is approximately horizontal or vertical, these prediction errors increase significantly. As the robot should keep a stable position in its whole workspace, the maximum of each joint error is chosen as a threshold. Table IV presents these thresholds for each joint and each payload.

*Evaluation by Visualization of Sensitivity:* To evaluate the quality of the interaction, the robot enters several fixed positions with a 1 kg payload. We measure the additional payload of the robot from which the robot starts updating its posture, as shown in Fig. 5a, 5b. Since this evaluation only covers forces in the direction of the gravity, we perform an additional evaluation with directional forces at the end-effector, see Fig. 5c, 5d. Due to the redundancy of the robot, different sensitivities for the same end-effector position can be achieved, as shown in Fig. 5.

*TABLE IV: Thresholds chosen for the Adaptive Assistance.*

| Joint $q_i$ | 0 kg | 1 kg | 3 kg | 5 kg |
|---|---|---|---|---|
| 1 | 4.5 | 5.2 | 5.2 | 6 |
| 2 | 2.7 | 3.5 | 5.5 | 6 |
| 3 | 1 | 1.2 | 2 | 2.5 |
| 4 | 0.5 | 0.7 | 1 | 1.3 |
| 5 | 0.3 | 0.4 | 0.6 | 1 |
| 6 | 0.2 | 0.2 | 0.3 | 0.5 |

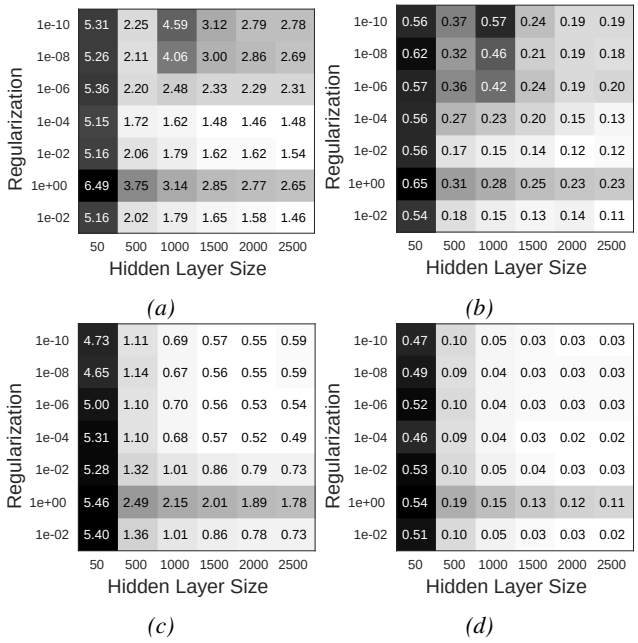



*Fig. 6: Optmization for torque and payload prediction for a single ELM model. The figure shows the cross validation results for torque prediction (left) of joint 2 to 5 for 1000 samples (a) and 7500 samples (c). In addition the cross validation results for payload prediction (right) are shown in (b) for 1000 samples and (d) for 7500 samples.*

### B. Variable Payload

The second part of the experiments focuses on the parametrization of the learned equilibrium model. As mentioned in Sec. III, we compare two methods of integrating the parameterization. For our experiments we use a variable load of the end-effector, but other continuous process parameter could be used as well, e.g. filling level of a bottle, material density or deformation. As previously, the parameters for learning the equilibrium model are optimized with a grid evaluation of a five-fold cross-validation. For the first method, we utilize a combined input space of joint angles and the payload at the end-effector to predict the required torques for each joint, as explained in Sec. III-A. However, when a robot handles an object or is in a certain configuration, the parameterization might be unknown. Therefore, we also evaluate the prediction of the model parameterization, i.e. payload. We do this by learning an ELM model with joint angles and torques as input and the current payload as output. For the second method we use a model space learning approach, therefore we train separate ELM models that map joint angles to torques for different load configurations. A second generalist ELM is trained to generalize between the readout weights of the torque prediction, as mentioned in Sec. III-A.

*Method I: Input Space Extension:* This algorithm uses the joint angles and the payload as input to predict the torque for a given robot configuration. The hidden layer is set to 1500 neurons and the regularization is set to $\lambda = 10^{-5}$. Like in the previous experiment, the result of the five-fold cross validation to estimate sufficient parameters can be found in Fig. 6a & 6c. To predict the payload, all six joint angles and six joint torques are used as input. As before the results of the five-fold cross validation can be found in Fig. 6b & 6d. The hidden layer is set to 2000 neurons and the regularization to

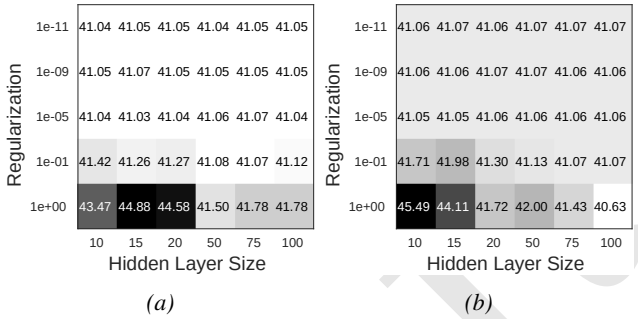



*Fig. 7: Parameter optimization for learning the model space parameters $W_{out}$ for 1000 samples (a) and 7500 samples (b).*

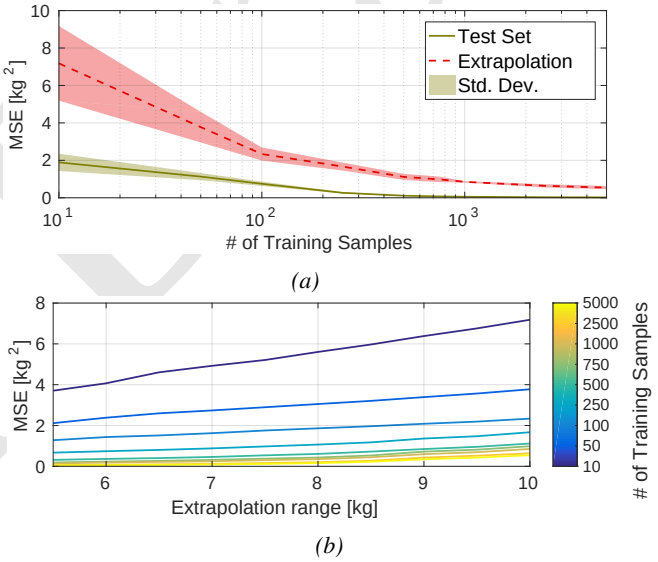



*Fig. 8: Evaluation of the payload prediction. (a) shows the MSE on the test set in relation of the number of samples from the training (green) and extrapolation (red). (b) evaluation of the extrapolation capabilities.*

$\lambda = 10^{-4}$. For training we used samples of our training data set with 0 kg, 1 kg, 3 kg, and 5 kg payload. The evaluation in comparison to the the model space learning approach is shown in Fig. 9(a-d), the evaluation of the prediction of the payload can be found in Fig. 8. It can be seen that the parameterized ELM model is able to reduce the prediction error in the case of torque and payload prediction significantly. In case of the extrapolation data set, it can be seen that the MSE of the torque prediction can not be significantly reduced by the ELM model (Fig. 9(e-h)). In case of the payload prediction (Fig. 8), the error is reduced as well although it can not reach the performance level as in the region of presented training data.

*Method II: Model Space:* For all samples of the training set with the same payload, a separate specialist ELM is

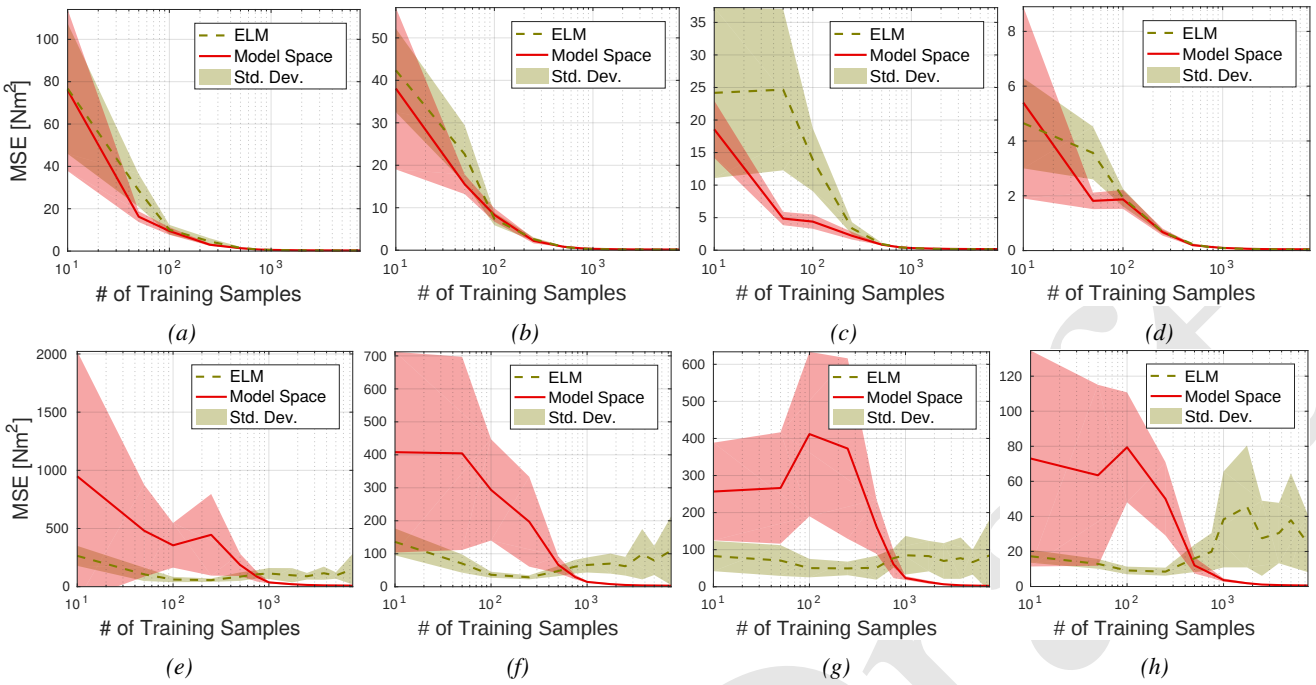



*Fig. 9: Evaluation of the parameterized torque prediction for joint 2-5 (from left to right). Comparison of input space extension (green) and model space learning (red): Test data set (top) and extrapolation data set (bottom). Mean and standard deviation of 10 trials.*

trained, with a hidden layer of 500 neurons and a regularization of $\lambda = 10^{-5}$. The randomly selected inputs weights $W_{inp}$ are shared to be able to generalize between the learner in the readout matrix $W_{out}$. To learn the mapping of context parameters (i.e. payload) and output weights $W_{out}$ of the specialist ELMs, a generalist learner, responsible for the model space, is used with a hidden layer size of 20 neurons and a regularization of $\lambda = 10^{-9}$. Again we estimated the parameters as in the previous case by five-fold cross validation as shown in Fig. 7. Since only a few data points for estimation of a mapping from payload parameterization to read-out weights are available, we expect high regularization and a low number of required hidden neurons. This assumption is confirmed as shown in Fig. 7, which shows the results of the mean squared error on the test set in relation to the parameterization of the learner. As shown in Fig 9(a-d), in comparison to method I the model space approach reaches comparable performance on the mean squared error for high amount of training data, the results indicate a slightly faster error decay, especially for training set sizes around 100 samples. For the comparison on the extrapolation data set, it can be seen that the model space learner has a higher variance for low amount of training data but is able to reach a better generalization for high number of training samples (Fig. 9(e-h)).

### C. Discussion

The simulation results suggest that the application of an equilibrium model for implementation of a compliant control mode is feasible for industrial lightweight robots. As shown in Sec. II, we are able to apply the equilibrium model for a range of different end-effector payloads. Beside the mean squared error on a distinct test set, we evaluated the sensitivity to external forces (Fig. 5) of the estimated model, sensitivity encodes which forces are necessary at the end-effector to trigger the adaptation of the robot's posture. By extending the equilibrium model with a parametrization for variable payloads, we have shown that a generalization of the predicted torques is possible for unobserved payload conditions. Additionally, we were able to predict the current payload by observing the joint position and torques (Fig. 8). We compared the combined input space with model space learning and we have shown that beside a slightly better performance during the learning process with small amounts of training data (Fig. 9), the model space learning has beneficial extrapolation capabilities, e.g. for high load conditions. As shown, the ELM of the combined input space is not able to generalize to the extrapolation data set, which can be caused by the higher number of hidden layer neurons.

## VI. CONCLUSION

We have shown the integration, implementation and evaluation of our adaptive handling system on an industrial lightweight robot. The implementation of the model learning algorithms and sampling of data sets were supported by a simulation based framework. However, the adaptation to external forces, by following their direction, is realized by utilization of a learned equilibrium model without the need to model the dynamics of the robot. This allows to cope with attached devices as well as devices with inaccurate dynamic models, e.g. stiff tubes, wires, protection shields or hose packages. The experiments demonstrate a successful parameterization of the equilibrium model and compare different learning concepts. Evaluation was conducted in

simulation and focuses on fixed as well as variable load configurations. Our results indicate that the proposed system can be applied in industrial use cases as well and we argue that a learning approach can be beneficial in highly flexible and uncertain application areas.

Future work includes the transfer of the results gathered by simulation to the real UR5 robot platform and evaluation of interaction with human users.

## ACKNOWLEDGMENT

J. Queißer received funding from the Cluster of Excellence 277 Cognitive Interaction Technology. Further, this work received funding from the German Federal Ministry of Education and Research (BMBF) within the Leding-Edge Cluster "Intelligent Technical Systems OstWestfalenLippe" (it's OWL) managed by the Project Management Agency Karlsruhe (PTKA).

## REFERENCES

[1] S. Popić and B. Miloradović, "Light weight robot arms - an overview," *INFOTEH-JAHORINA*, vol. 14, 2015.

[2] N. D. Vuong and M. H. A. Jr., "Dynamic model identification for industrial robots," *Acta Polytechnica Hungarica*, 2009.

[3] S. Goto, N. Nakamura, and N. Kyura, "Forcefree control with independent compensation for inertia friction and gravity of industrial articulated robot arm," in *IEEE International Conference on Robotics and Automation*, vol. 3, Sept 2003, pp. 4386–4391 vol.3.

[4] L. Ding, H. Wu, Y. Yao, and Y. Yang, "Dynamic model identification for 6-dof industrial robots," *Journal of Robotics*, vol. 2015, p. 9, 2015.

[5] J. F. Queißer, K. Neumann, M. Rolf, F. R. Reinhart, and J. J. Steil, "An active compliant control mode for interaction with a pneumatic soft robot," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Chicago, IL, 2014, pp. 573–579.

[6] S. Wrede, O. Beyer, C. Dreyer, M. Wojtynek, and J. J. Steil, "Vertical Integration and Service Orchestration for Modular Production Systems using Business Process Models," 2016.

[7] P. Tomei, "Adaptive pd controller for robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 4, 1991.

[8] H. Berghuis, R. Ortega, and H. Nijmeijer, "A robust adaptive robot controller," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 6, 1993.

[9] S. S. Ge and C. C. Hang, "Direct adaptive neural network control of robots," *Int. Journal of Systems Science*, vol. 27, no. 6, 1996.

[10] S. S. Ge, C. C. Hang, and L. C. Woon, "Adaptive neural network control of robot manipulators in task space," *IEEE Trans. On Industrial Electronics*, vol. 44, no. 6, 1997.

[11] I. Ranatunga, S. Cremer, F. L. Lewis, and D. O. Popa, "Neuroadaptive control for safe robots in human environments: A case study," in *IEEE Int. Conf. on Automation Science and Engineering (CASE)*, 2015, pp. 322–327.

[12] A. D. Luca and S. Panzieri, "A simple iterative scheme for learning gravity compensation in robot arms," in *36th ANIPLA Annual Conf. (Automation*, 1992, pp. 459–471.

[13] A. De, Luca, and S. Panzieri, "Learning gravity compensation in robots: Rigid arms, elastic joints, flexible links," in *Int. Journal of Adaptive Control and Signal Processing*, 1993, pp. 417–433.

[14] J. Swevers, W. Verdonck, and J. D. Schutter, "Dynamic model identification for industrial robots," *IEEE Control Systems*, vol. 27, no. 5, pp. 58–71, Oct 2007.

[15] M. Rolf, K. Neumann, J. F. Queißer, F. R. Reinhart, A. Nordmann, and J. J. Steil, "A multi-level control architecture for the bionic handling assistant," *Advanced Robotics*, vol. 29, no. 13, 2015.

[16] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme Learning Machine: a New Learning Scheme of Feedforward Neural Networks," in *IEEE Proc. Int. Joint Conf. on Neural Networks*, 2004, pp. 985–990.

[17] W. Deng, Q. Zheng, and L. Chen, "Regularized extreme learning machine," *IEEE Symp. on Comp. Intelligence and Data Mining*, 2009.

[18] W. Aswolinskiy, F. Reinhart, and J. J. Steil, "Modelling of Parameterized Processes via Regression in the Model Space," in *Proc. of 24th European Symposium on Artificial Neural Networks*, 2016, pp. 53–58.

[19] *Gazebo Robot Simulation Tool*, (accessed on August 29, 2016), http://gazebosim.org.

[20] *Orocos - Open Robot Control Software*, (accessed on August 29, 2016), http://www.orocos.org/.

[21] *Orocos RTT - Real-Time Toolkit*, (accessed on August 29, 2016), http://www.orocos.org/rtt.