To appear in *Advanced Robotics*
Vol. 00, No. 00, January 2015, 1–15

FULL PAPER

A Multi-Level Control Architecture for the Bionic Handling Assistant

M. Rolf[a,*], K. Neumann[b] J. F. Queißer[b], R. F. Reinhart[b], A. Nordmann[b], J. J. Steil[b]

[a]*Department of Adaptive Machine Systems, Graduate School of Engineering, Osaka University, Japan*;
[b]*Research Institute for Cognition and Robotics (CoR-Lab), Bielefeld University, Germany*

()

The Bionic Handling Assistant is one of the largest soft continuum robots and very special in being a pneumatically operated platform that is able to bend, stretch, and grasp in all directions. It nevertheless shares many challenges with smaller continuum and other softs robots such as parallel actuation, complex movement dynamics, slow pneumatic actuation, non-stationary behavior, and a lack of analytic models. To master the control of this challenging robot, we argue for a tight integration of standard analytic tools, simulation, control, and state of the art machine learning into an overall architecture that can serve as blueprint for control design also beyond the BHA. To this aim, we show how to integrate specific modes of operation and different levels of control in a synergistic manner, which is enabled by using modern paradigms of software architecture and middleware. We thereby achieve an architecture with unique overall control abilities for a soft continuum robot that allow for flexible experimentation towards compliant user-interaction, grasping, and online learning of internal models.

Keywords: Continuum Robot, Soft Robot, Control Architecture, Middleware, Compliant Interaction

1. Introduction

In recent years, an increasing number of continuum robots have surfaced in various forms and fields. Prominent examples include artificial salamanders [1], hexapods [2], snakes [3], worms [4], and smaller quadrupeds [5]. These platforms showcase the interplay of morphology and computation [6] and explore the benefit of highly flexible continuum robots for future applications, like minimal invasive surgery [7]. They also provide a way to implement the understanding-by-building paradigm towards analysis of biological systems, e.g. to understand an octopus tentacle [8] or movements of a fish [9].

The Bionic Handling assistant (BHA, Fig. 1 left) has been designed by Festo as a robotic counterpart to an elephant trunk. It has gathered strong interest in the robotics community as well as the general public because it belongs to a new class of continuum soft and lightweight robots based on low-cost and rapid 3D manufacturing with polyamide. It comprises several continuous parallel components and is pneumatically operated at low pressures, which makes the BHA inherently safe for physical interaction with humans [10, 11]. The BHA's main body embodies three segments[1], each consisting of three triangular arranged air chambers, i.e. in total nine. These extend in length relative to the applied pressure. The robot has therefore no fixed joint angles and each segment rather starts to bend whenever the three chambers assume different

*Corresponding author. Email: matthias@ams.eng.osaka-u.ac.jp
[1]An additional gripper segment is also available, but neglected for this work

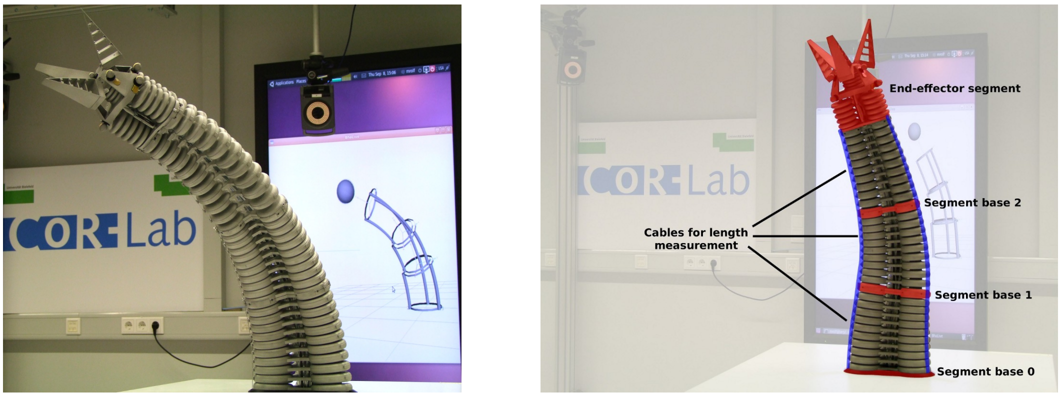

Figure 1. The bionic handling assistant BHA (left). The segments and respective length sensors of the BHA (right).

lengths. An active depression of the chambers is not possible. Solely the internal tension of the extended body drives the structure back into its default shape. The BHA is equipped with pressure sensors inside the air valves. Potentiometers inside the base measure the lengths of cables across the outer robot structure and therefore provide geometric information about the robot's shape (Fig. 1, right).

Up to the presented work, no comprehensive automatic control has been introduced for the BHA despite the big potential of its unprecedented movement flexibility. The reason is likely that the BHA comprises substantial challenges for any control scheme including high dimensionality and redundancy, very slow actuator dynamics, restrictive and unknown actuation ranges, and non-stationary system behavior due to friction and visco-elasticity.

Related work that tackles the control of soft robots mostly focus only on single aspects rather than complete control architectures like learning the dynamics of an octopus tentacle [12] or even fall back to manual control [13]. These robots share a lot of challenges with the BHA, e.g. Shepherd emphasized that "... the response to actuation of elastomeric structures having embedded PNs[1] is highly nonlinear and thus predictive modeling of their actuation is currently empirical. The development of motion control systems for these robots will require the use of nonlinear models and may require neural-net-like learning methods." ([13], p. 20403). And in fact, it was shown in other work that for modeling the inverse kinematics of soft robots, machine learning can be beneficial in comparison to classical Jacobian gradient based methods [14].

While these studies address isolated control problems of soft robots, no comprehensive approach towards entire architectures that combine several control skills and facilitate the development of high level use cases has been presented so far. The contribution of this paper is to tackle this architectural problem. Referring to our prior work on isolated problems on the BHA [15–17], we now describe the challenges induced by the special hardware properties of soft robots and argue that well designed software architectures are indispensable for effective implementations of higher level functionality. We therefore show how modern software engineering paradigms allow for flexible experimentation with low-level components and higher-level use cases and provide a step forward to leverage the full potential of continuum robot applications. We argue and show that a carefully selected hybrid combination of classical control methods and machine learning throughout control levels can achieve fast reaction times and high precision for real world applications. Further, we use this architectural perspective to point out synergy effects between different control skills that amplify the utility of those skills within a larger context. While we demonstrate our proposal based on the BHA, we also believe that many of the underlying elements are applicable to other robots as well. After describing the control architecture and some of the functional building blocks, we will come back to this issue and finally elaborate on general conclusions.

[1] Pneumatically Actuated Pneumatic Networks (pneu-nets) [13]

2. Overview of the control architecture

2.1 *Challenges and requirements*

The Bionic Handling Assistant (BHA) is a prominent, award winning[2], continuum soft robot. It displays typical challenges in soft robotics. The most significant challenge is induced by its novel actuation principle of co-activaion of three low-pressure pneumatic actuators in each segment that cause continuous deformations in shape. This is complementary to revolute or prismatic joints that drive classical robots. Classical rigid body mechanics and respective control schemes can therefore not be transferred easily from traditional robotics. Neither standard serial kinematic chains, nor parallel but rigid mechanisms model the BHA well. In the BHA, continuous deformations are caused by parallel actuation, but mediated through flexible material and morphology in a highly nonlinear and difficult to predict manner. This setting calls for new and advanced algorithms to cope with the resulting redundancy, with non-stationarity due to the semi-fluid properties of the material, and the slow dynamics of the pneumatic actuation.

More technical, but for soft robotics very typical challenges include the lack of software infrastructure for control, simulation, or even task-level operation. It is neither provided by the producer FESTO nor does the futuristic and experimental hardware allow application of standard tools. For instance, due to the lack of a kinematic description in standard DH-parameters, no off-the-shelf simulation and visualization tools can be used. On the hardware level, the BHA provides heterogeneous I/O channels combining pressure sensing and control via a CAN bus with length sensing via an analog-digital converter PCI card. In our lab, we also integrate external end-effector position sensing via a VICON motion tracking system that communicates with yet a different proprietary network protocol. Finally, beyond just controlling the robot we target to leverage the full potential of the BHA for safe interaction with humans in different use-cases and applications which request stability, robustness and repeatability of experimentation.

The challenges translate to a multitude of requirements for a comprehensive control architecture for the BHA. It needs: abstraction of sensory data sources of different temporal resolution that are read from hardware sensors or internal models; different hardware abstraction levels that include pressure control, posture control, and end-effector control; the integration of multiple models that depend on each other and share data across several abstraction levels; the hierarchical combination of controllers over several layers of abstraction; realtime and online capabilities for interactive scenarios; high modularization to allow a flexible reconfiguration for different use cases. We address these requirements in form of the control architecture depicted in Fig. 2 and by means of efficient software and middleware tools realizing the control flow, the structural modules and their communication, possibly across different processes.

2.2 *How to Control the Bionic Handling Assistant*

All components of the control architecture are assigned to three levels. The bottom level (Fig. 2, bottom) is related to hardware specific implementations and the BHA robot itself. This level is described in detail in Sect. 3.1. It schematically visualizes the implementation details related to the hardware protocols (denoted with *Real Plant*) and the kinematic simulation of the BHA (denoted with *Virtual Plant*). The involved components of this level are hardware-specific, deal with the peculiarities of proprietary software modules, and can be distributed across several computing machines. The next level (Fig. 2, middle) provides a robot specific interface and hides the complexity of the underlying low level by means of hardware abstractions. It is further described in Sect. 3.2. A central component of this level is the *Control Interface Library* that allows both a seamless swapping between the simulation and the hardware of the BHA and a parallel operation as use cases require. Important roles play the learned *Inverse Equilibrium*

[2]BHA won the prestigious German "Zukunftspreis" (future award) in 2010.

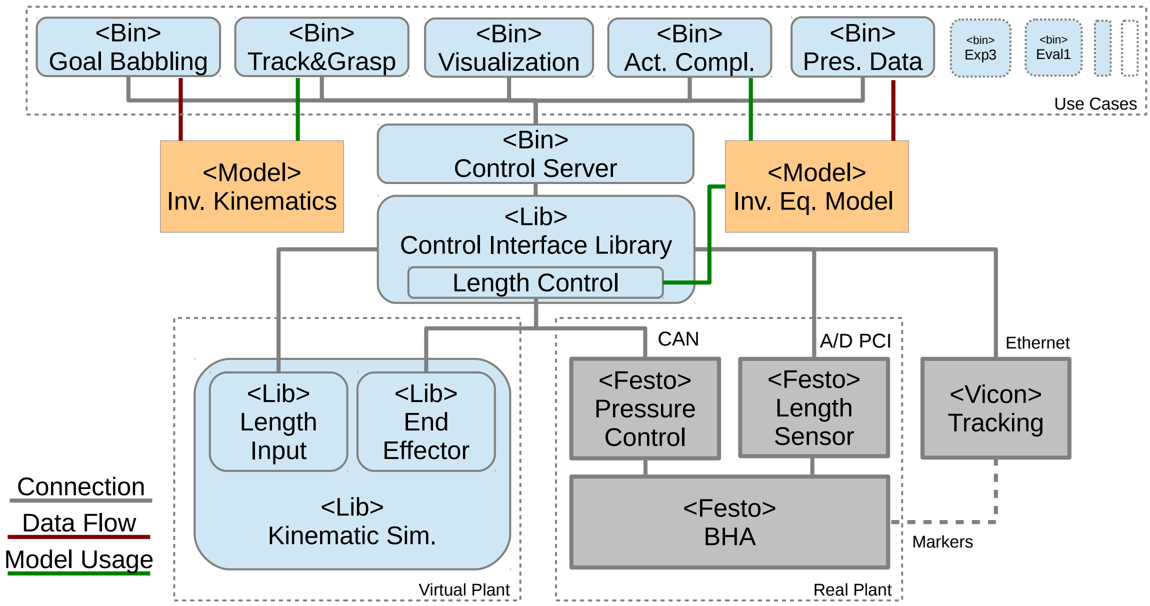

Figure 2. Schematic view of the system architecture. The BHA (real plant) and its kinematic simulation (virtual plant) are connected via the control interface library to the use cases. This software and system architecture provides a profound basis for experimentation and development of diverse applications.

Model (IEM) and the learned *Inverse Kinematics Model (IKM)*, as shown in Fig. 2. They are kept externally and can be loaded at runtime, which makes them available in higher and lower level components and allows for offline or online learning. This renders the overall architecture much more flexible and enables to utilize hybrid control schemes across levels and applications. The top level of abstraction contains the software components representing the actual *Use Cases* for experimentation and application development, see Sect. 5. They communicate with the unified interfaces at the second level and are thereby independent of hardware details, can flexibly recruit and load learned models, and deploy the actual BHA, the kinematic simulation, or both. In Sect. 5, we elaborate some of our more complex applications comprising *Visualization* (Sect. 5.1), an *Active Compliant Control Mode* (Sect. 5.2), and the exploration and learning of an *Inverse Kinematics Model* by means of Goal Babbling [16].

2.3 *Software architecture and middleware*

The overall architecture is modular and relies on software abstractions to facilitate software and application development. It uses the *Robot Control Interface* (RCI) [18] which provides a set of domain-specific abstractions to represent common features of compliant robotics systems. These abstractions comprise for instance a synchronized robot interface that supports sensory readout and command transmission. The flexibility of RCI, which abstracts from the low-level signals, is also used to bundle actuators segment-wise, as the elongation of one single actuator is meaningless for the posture of one segment (see Fig. 1).

For implementation of functional components and integrating RCI entities into the applications, we leverage the *Compliant Control Architecture* (CCA) [18]. CCA is an event-based, middleware-agnostic component architecture for robotics research and focuses on control of compliant hardware and machine learning. The library serves as technology mapping for platforms modeled in RCI and as component architecture for implementing applications. This leads to independent software components that can be flexibly combined in component circuitry. Communication between components is realized by the *Robotics Service Bus* (RSB) middleware [19]. RSB is very lightweight, fast, and can be used with a very small footprint of source code, i.e. it does not create a significant “lock-in” of source code that would prevent using the same code without RSB.

There are several advantages of this strategy: processing-intensive applications can be distributed; a central software instance to operate the robot can be running while other applications (high-level applications, visualization, logging) can be turned on and off flexibly as it is needed; the network interface permits vast changes of the implementation without requiring other components to change. This backbone of software engineering provides the basis for the realization and seamless integration of the functional blocks that are shown in Fig. 2 and described below. Due to the modular structure of the software framework, each component of the low-level control is available by a unified interface and can be reused for other control modules as required. This is demonstrated in the *Use Case* scenarios in Sect. 5. We proceed by describing in more detail the functional components in the different levels of the architecture and their interactions.

3. Functional building blocks

3.1 *Bottom Layer and Hardware Abstraction*

The lowest level of the system architecture provides an abstraction of the underlying hardware. It allows to control the actuators and to access the sensors without knowledge about the specific hardware implementation. It thereby provides the means to address the BHA *(Real Plant)* and its kinematic simulation *(Virtual Plant)* through identical interfaces.

3.1.1 Real Plant

This functional module deals with the immediate actuation and sensors of the actual BHA. The only immediate actuation on the BHA is pressure control, which is integrated in two valve units manufactured by Festo. They comprise eight compact Piezo valves each and control the BHA's pneumatic actuators. Both valve units with their pressure controllers are connected to a PC via CAN-Bus with a proprietary protocol. Our software abstracts from that CAN interface and provides generic methods to read current pressures and set control targets.

The length sensors that sense the BHA's shape can be read via an analog/digital PCI card with high frequency. Since the cables span the entire length from the BHA's base to the end of each actuator (see Fig. 1), the values along the robot are automatically subtracted to obtain the outer length of each separate actuator. The sensor readings are Kalman-filtered for noise reduction before being dispatched through the software. Additionally, we use an external Vicon motion tracking system [20] to determine the 3D spatial location of the robot's end-effector. The Vicon software [20] sends data with 200Hz via a proprietary data protocol on a TCP network connection. In order to utilize this data, we use a converter that broadcasts the data via the RSB middleware in a computationally very efficient binary format.

3.1.2 Virtual Plant

An essential part of the overall system is a useful kinematics model in order to support higher level use cases for the BHA robot is a useful kinematics model. However, the simulation of a continuum robot such as the BHA cannot be solved by classical approaches and standard software tools cannot be applied. While a model of the BHA dynamics is clearly out of reach, we have developed in previous work [15] an approximate kinematic model ignoring pressures and solely operating on the lengths of virtual air chambers. To this aim, we followed the constant curvature approach that is based on torus segments in order to allow continuous deformations. Fig. 1 (left) shows how the three actuators in each main segment cause a deformation between two rigid segment bases (shown in red). The three measured lengths of these actuators can be used to estimate the coordinate transformation between two platforms, which can then be chained in order to get the complete forward kinematics from base to end effector. For each particular deformation in three dimensions, a segment of the robot can be modeled by a torus segment. Fig. 3(left) shows this relation with the overall torus in light green and the robot segment in dark green. The only free parameters of the segment model are the segement radii.

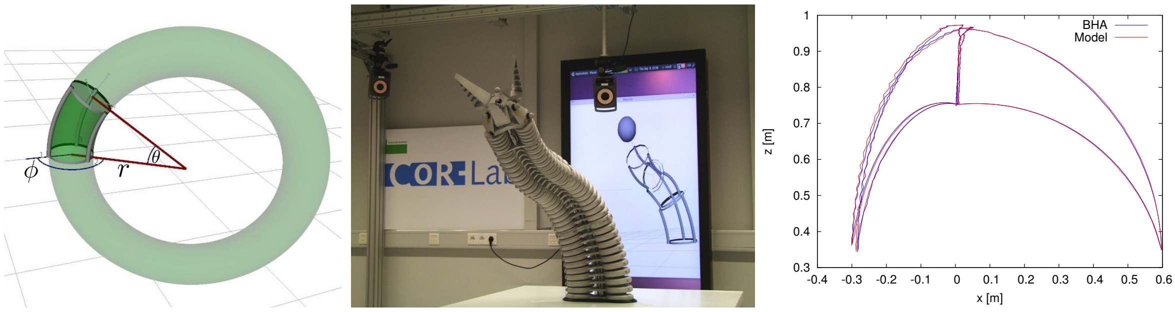

Figure 3. Kinematic simulation: Segment model including geometric parameters (left), posture example along with visualization based on the simulation model (center), and evaluation of the model quality for exemplary movements (right).

We estimated these radii according to a best-fit solution by recording ground-truth data from different movements.

The development of the kinematic model is required for a 3D visualization of the robot, however, without millimeter accuracy. To this aim, length measurements from the BHA can be taken to predict and visualize the Cartesian movement of the robot (Fig. 3 (center)). Our implementation of the constant curvature model including the visualization is available as open source[1].

3.2 *Medium Level Control Interface Library*

The core element of the control architecture is the *Control Interface Library*, as shown in Fig. 2. The library unifies the interfaces of BHA simulation and BHA hardware and provides the basic control algorithms. It provides the sensing of the current lengths L^{real}, pressures p, predicted pressures for target lengths $\hat{p}$ and the end-effector position EE relative to the robot base. Based on the abstractions to access the hardware, we can command target pressures $p^{des.}$ to Festo's valve units as most direct form of control. Still, pressures only describe forces acting on the robot structure which does not allow for a robust postural control of the robots shape. Hence, it is pivotal to actively control the posture of the robot, i.e. the lengths of the actuators through commanding target lengths $L^{des.}$. This invokes either a standard PID length controller or a more sophisticated hybrid length controller, which additionally employs an external and learned feedforward model. This controller is described in more detail in Sect. 4. Each high level module of the framework integrates the *Control Interface Library* and is able to connect to the underlying setup via the *Control Server*, given in Fig. 2.

The *Control Interface Library* is pivotal to the overall architecture by fully utilizing the power of the RSB middleware. It automatically instantiates an RSB structure called *Informer* that broadcasts information throughout the network to all components and applications that have subscribed to it. The library broadcasts all available sensory data, as well as all motor commands currently active on the robot. Since this is fully automatic, the information is available in the middleware whenever and however the robot is operated. Hence, tools e.g. for logging or visualization can be started and stopped at any time. Additionally the robot's low-level control can be accessed via the *Control Server* that receives commands via RSB. The entire broadcast as well as the control server utilize a flexible text/XML format for exchange. This allows to distinguish semantically different values of the same measure (e.g. measured vs. desired) and to fluently add additional information.

[1] http://www.cor-lab.org/software-continuum-kinematics-simulation

3.3 *External Models and Learning*

External models play a special role in our architecture. As emphasized before, there is no simple way to model actuators, kinematics or dynamics in classical terms or with standard tools. We therefore opt to apply machine learning methods, as proposed also in [13] and much of our previous work [16, 17, 21, 22], in particular for learning internal models. We keep these models separate and loadable, as they may be *re*-trained, *re*-calibrated or even *re*-learned online. The models can then be utilized in several parts of an application. Two such models, which are of crucial importance in any control architecture, are depicted in Fig. 2. On the right hand side, an inverse model for the actuator dynamics is pre-learned offline as described in more detail in Sect. 4 because of its importance even for the lower-level length control. From the point of view of architecture design, it is important to see that this model is exploited from the *Control Interface Library* to operate in parallel to the basic PID control loop to speed-up pneumatic actuation. Yet, it is as well used in the compliant control application as detailed in Sect. 5.2.

In a similar vein, a learned inverse kinematics model is displayed on the left hand side of Fig. 2, which can be loaded and exploited, e.g. for grasping applications. But it can also be explored, learned and modified online, for instance to cope with a changing redundancy resolution that is necessitated by the non-stationarity of the BHA [16]. Again, the software architecture allows for the flexible use of this model and its embedding in different applications. Both non-stationarity and redundancy resolution are typical problems for soft robots, which often require to apply learning methods and therefore we believe that the solution to separate these models from the control library could be useful also for other platforms.

3.4 *Higher-Level Control and Applications*

The tools introduced so far provide the necessary components for application development. Three such applications will be functionally described in detail in Sect.5. Applications and use cases can flexibly recruit components, models, the hardware and the simulation, and all quantities broadcasted by the control library. Relying on the middleware communication, an arbitrary number of such applications can run simultaneously, for instance to perform high level tasks and experimental evaluations. Components that implement task-specific behaviors can be connected or disconnected at runtime. Fig. 1 and 3 already emphasize this flexibility by showing the combination of a basic control loop running on the real BHA and the visualization of the robot in the background through the kinematic simulation. In this case the real and the virtual models are running in parallel.

Human-Robot-Interaction, grasping tasks and other applications require an interaction with the physical environment and rely on realtime capabilities of the underlying control framework. Although specific scenarios typically involve many components, the utilized software architecture is able to deal with the data flow in real time. The timed *Control Interface Library* can be configured with millisecond precision to satisfy control and platform demands. For our control tasks a high-level discretization time frame of $20ms$ is sufficient. The distribution of components over several computing machines, which is supported by the middleware RSB [19], can contribute to the relaxation of temporal constraints.

4. Hybrid Actuator Dynamics Control

A reliable and fast controller of the BHA's actuator lengths that determine the robot's shape is an indispensable prerequisite for higher level control skills and applications. In principle, the length control can be accomplished with standard proportional integral derivative (PID) schemes. The fundamental problem is that these feedback control approaches can be applied only with low gains in case of slow plant dynamics, which consequently results in very slow movements. This is in particularly the case for the BHA due to its pneumatic actuation and the visco-elastic mechanics, however, the BHA shares this issue with many other soft robots with pneumatic or cable driven elastic actuation.

A classical approach to achieve fast control in such cases is to combine feedback and feedforwad control. The feedforward controller provides anticipative commands and can significantly reduce control delays. However, feedforward requires an inverse model of the plant's dynamics. For the BHA, such an inverse model would map actuator lengths $\mathbf{l}$ and their derivatives $\dot{\mathbf{l}}$ and $\ddot{\mathbf{l}}$ to pressures $\mathbf{p}$ and pressure changes $\dot{\mathbf{p}}$ in the actuators according to the pneumatic dynamics [23]:

$$\dot{\mathbf{p}}(t) = \mathbf{f}(\mathbf{l}(t), \dot{\mathbf{l}}(t), \ddot{\mathbf{l}}(t), \mathbf{p}(t)). \tag{1}$$

Although the inverse model does not have to be very accurate because the feedback part of the control law compensates errors, there is no analytic model available for the soft BHA. This context qualifies learning as an essential tool for modeling.

4.1 *Fast Control with an Inverse Equilibrium Model*

The fundamental challenge for learning is the generation of sufficient data. Eq. (1) describes very high dimensional interactions that can not be fully explored on the real robot because of the exponential increase of exploration costs with the increasing dimensionality of the configuration space, i.e. here the lengths. We therefore consider a simplified model of the robot's dynamics, which is restricted to the mechanical equilibrium points $\mathbf{l}^*$ of the robot's dynamics. Equilibrium points are achieved by applying a constant pressure $\mathbf{p}^*$ until convergence of the lengths for a single segment. In such equilibrium states, neither lengths nor pressures of the pneumatic actuators change over time: $\dot{\mathbf{p}} = \dot{\mathbf{l}} = \ddot{\mathbf{l}} = 0$. The formulation of the inverse dynamics in Eq. (1) then simplifies to

$$0 = \mathbf{f}(\mathbf{l}^*, 0, 0, \mathbf{p}^*) \;\Leftrightarrow\; \hat{\mathbf{p}}(\mathbf{l}^*) = \mathbf{p}^* \;, \tag{2}$$

where $\hat{\mathbf{p}}$ denotes the inverse equilibrium model that represents the direct relation between length $\mathbf{l}^*$ and pressures $\mathbf{p}^*$. The inverse equilibrium model provides a direct estimation of air chamber pressures in a mechanical equilibrium and can therefore serve as a feedforward control signal. Fig. 4 depicts the BHA plant with its slow dynamics, the low-gain PID feedback controller, and the inverse equilibrium model. The BHA receives pressure commands, which are computed by superimposing the PID and the feedforward control signals. The feedforward controller computes pressures from desired lengths by means of the inverse equilibrium model. PID control is based on the difference of the desired and sensed length values. The PID controller corrects errors of the feedforward control signal in the feedback loop.

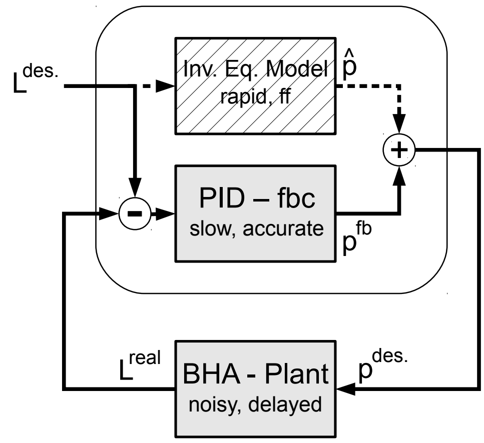

Figure 4. Control loop with a learned inverse equilibrium model and a feedback controller. The model leads to a fast estimation of the pressure configuration p^{des} for the chamber lengths L^{des}.

4.2 *Learning an Inverse Equilibrium Model*

Despite these simplifications, the learning is still difficult: first, data sampling for learning is limited because the time until the physical deformations of the robot have reached a mechanical equilibrium can take up to 20 seconds for a single data point. Second, the underlying dynamics of the BHA result in non-linear behavior which requires a model with appropriate complexity in order to capture the structure of the data sufficiently. Third, data is very noisy due to hysteresis effects induced by the visco-elasticity of the robot's soft material.

Machine learning approaches which are trained on such sparse and noisy data without additional efforts are prone to overfitting, whereas well-behaved extrapolation is a strong requirement from the BHA. To achieve this generalization from sparse data, we employ the constrained ELM, which is able to embed prior knowledge about the physical behavior of the BHA to prevent strong overfitting. The learning scheme is called Constrained Extreme Learning Machine (CELM, [24]) and comprises a feedforward neural structure with three layers of neurons. Due to the special form of this approach, learning reduces to a linear optimization problem and the prior knowledge can be incorporated by introducing linear inequalities to the optimization program. For learning an inverse equilibrium model of the BHA, the following prior knowledge is considered: (**i**) maximum and minimum pressure of the actuators, and (**ii**) that the ground-truth behavior per axis is strictly monotonous, because higher pressure in one actuator physically leads to an extension of this actuator.

4.3 *Experimental Results*

For training of inverse equilibrium models, a data set of pressure-length pairs is recorded. It captures the relation between the geometric length of the actuators for each segment and the corresponding pressures in a mechanical equilibrium. A pressure grid comprising $5 \times 5 \times 5 = 125$ samples repeated five times is available for learning. Experiments on the robot show the benefits of the learned inverse equilibrium model for length control. For a quantitative evaluation, we measure the time until convergence of the lengths to different target values up to accuracy ε. Fig. 5 shows the mean convergence time for repetitively approaching five random length configurations with the BHA. Length control with simple PID control requires a much longer convergence time than with CELM model.

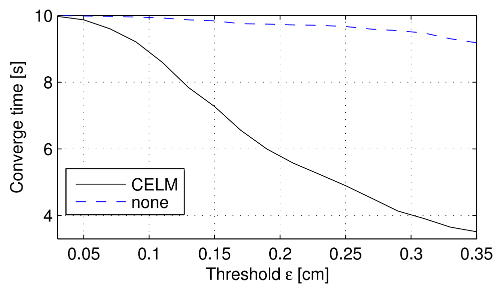

Figure 5. Convergence time of two different length controllers: simple PID (none) vs. feedforward control with inverse equilibrium model (CELM).

Note that the data recording and substitution of different models for the length control is strongly supported by the system architecture. Learning an inverse equilibrium model is essential for agile motion control of the BHA and represents a building block that is heavily used by other BHA applications (see Fig. 2).

5. Use Cases

This section exhibits three exemplary use cases as depicted in the top level of Fig. 2. These illustrate how to orchestrate modularization, interface unification, data sharing and extensibility towards real applications. Use cases address for instance high level control or task learning, novel technology demos, exploration of novel learning schemes, or user evaluation and can be easily developed and integrated into the framework. Thereby the development of new use cases can utilize the overall well-structured software organization together with the development tools offered by the middleware and component architecture.

5.1 *Use Case 1: Visualization*

One of the simpler but most prominent use case is the visualization of the BHA's current state and its control values. It illustrates a central element of our overall strategy how to perform development and experimentation on this platform. The basis for the visualization is the constant curvature model of the BHA as mentioned in Sect. 3.1.2. A torus model is used to render each segment of the robot (based on the current length measurement) in a 3D environment. The same environment can also be used to render various objects or coordinates that arise in other use cases such as learning (see Sect. 5) or object manipulation. Additional windows show the pressure and length values of the nine main actuators. If a motor command is active on any of those, the "raw" target as well as its filtered and clamped value is shown in relation to the measured sensory values. The utility of this tool largely roots in the automatic creation of a RSB Informer and the connection to the *Control Server* (see Fig. 2) by any BHA application as previously described. Due to this informer the to be displayed values are available within a unified interface supported by the middleware whenever a BHA application is running. The visualization tool hence can be started and stopped as a separate process when it is needed and there is no need to restart the visualization when the BHA application is stopped or restarted.

Fig. 3 (center) shows an example posture of the robot during dataset recording and its visualization generated by the constant curvature model. Note that this experiment underlines the effectiveness of how we integrated the kinematics simulation into the system architecture: the visualization actually requires to run the virtual plant, i.e. the kinematic simulation, as a parallel and independent module, which estimates the most likely posture with regard to the data from the BHA in real time and then feeds the graphical interface with the simulated posture.

The evaluation [15] of the model on the example data in Fig. 3(right) shows the comparison of ground-truth data (blue) and model predictions (red). With the estimated segment radii, the average accuracy is $0.0102m$. Related to an approximate (average) robot length, this corresponds to a relative error slightly above 1%. Neither the assumption of circular shapes, nor the assumption of equal radii within segments hold exactly on the real robot. However, it is noteworthy to see that the model reaches 1% relative error, while constant curvature models have dramatically failed for other robots, and even expensive, 'geometrically exact' models have only reached $1.5 - 5\%$ relative error [25], [26]. The 1% error holds only for the prediction of postures that are known to be possible (e.g. actually measured). We found that prediction from 'hypothetical' postures is much more difficult because it needs to be tested whether they are in range, and need to be projected to the closest possible one if they are not. Due to the specific geometry of the BHA even slight errors in this process can cause large mispredictions, e.g. $10cm$ sideward deviation of the end-effector for only $1cm$ mispredicted ranges on the lowest segment, see also the details in [15].

5.2 *Use Case 2: Actively Compliant Actuator Control*

To leverage the potential of the BHA for human-robot interaction, kinesthetic teaching, i.e. physical guidance of the BHA towards desired postures is a means of choice. In comparison to kinesthetic teaching on stiff robots [27–29], a flexible robot structure as in the BHA allows for mechanical deformation of its body due to its softness. The detection of a deformation, e.g. caused by a human tutor, can then be utilized to initiate a modification of the control variables such that the robot complies with the deformed configuration by actively controlling its actuators. We have shown that this idea can be used to implement an active compliance control mode without explicit force sensing [17]. To achieve this goal, the learned inverse equilibrium model of the robot is used to detect deflections from the equilibrium by comparing the measured pressures of the chambers with the expected chamber pressures for the current lengths. The control target lengths are then adjusted accordingly such that the current configuration becomes the new equilibrium point of the robot. This morphology-driven external force detection principle reduces the required computational effort and control complexity in comparison to classical approaches based on a full inverse dynamics model and accurate force sensing.

Fig. 6 shows the interconnection of the active compliance mode application with the previously described system infrastructure. The figure highlights additional software components in red while the already existing components are depicted in gray. The application requires a realtime

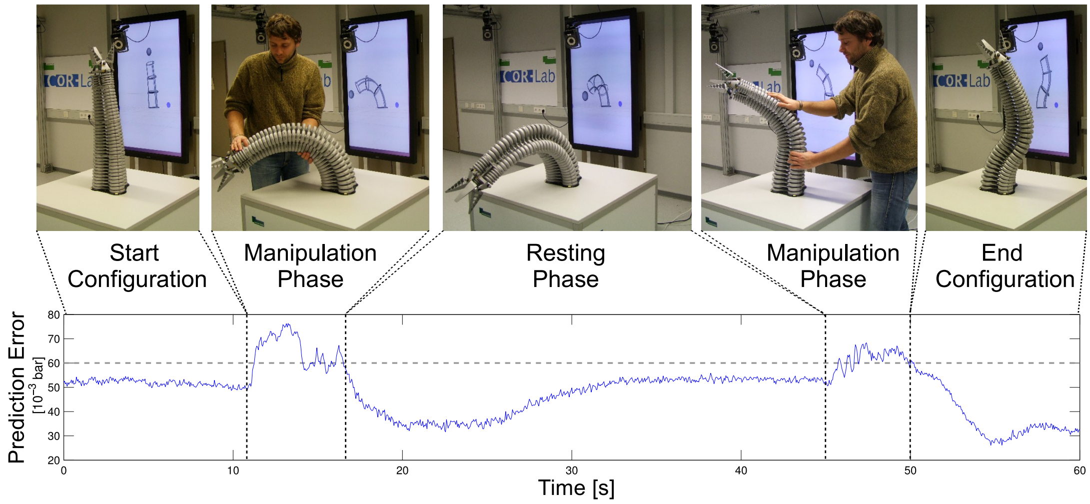

Figure 7. Active posture control in human-robot interaction. The graph on the bottom shows the prediction error $\|\mathbf{p} - \hat{\mathbf{p}}\| / \dim(\mathbf{p})$ between the actual and estimated pressure during human-robot interaction. The dashed line marks the threshold T. The prediction error exceeds T during the manipulation phase and falls below T during the resting.

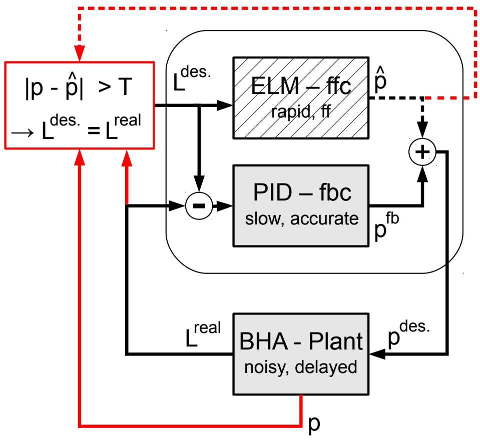

Figure 6. Active compliant control mode of the BHA achieved by application of a learned inverse equilibrium model of the pressure-to-length relation in a mechanical equilibrium.

communication pipeline to the inverse equilibrium model, because comparisons between measured $\mathbf{p}$ and predicted pressures $\hat{\mathbf{p}}$ according to the learned model must be performed online. The integration of this control mode into the software framework benefits from the high degree of modularization and the common interfaces unified by the *Control Interface Library* (see Fig. 2). Furthermore, it re-uses the external model and therefore minimizes the need for redundant implementation of certain models.

Fig. 7 shows the actual use case in a sequence of two manual reconfigurations of the BHA by a human tutor. The start configuration of the BHA is relaxed, the pneumatic actuators are deflated. After eleven seconds, the human operator starts to push the robot to the right side which deflects the robot's state from the mechanical equilibrium point. This instantly induces an increasing prediction error (see Fig. 7, bottom) as the actual pressure and predicted pressure to not coincide anymore. When the error exceeds a given threshold, the set-point of the length controller is updated to the current length sensor values. The length controller then adopts the pressures accordingly such that the current robot configuration becomes the new equilibrium point of the system. This tracking of the robot posture enables the user to easily change the posture of the BHA. After a short time span, the robot again reaches a mechanical equilibrium such that the error falls below the threshold. During this time, the arm stays fixed until a second manipulation phase is started by the user. The manipulations ends after the desired end posture is reached. The BHA stably stays in this position.

The experiments show that the proposed system architecture allows an easy implementation, execution, and testing of an active compliance mode that deals with the realtime constraints of kinesthetic teaching without the need of complex internal models of the actuator. Such human-robot interaction modes offer new fields of application for continuum robots in research and practical applications.

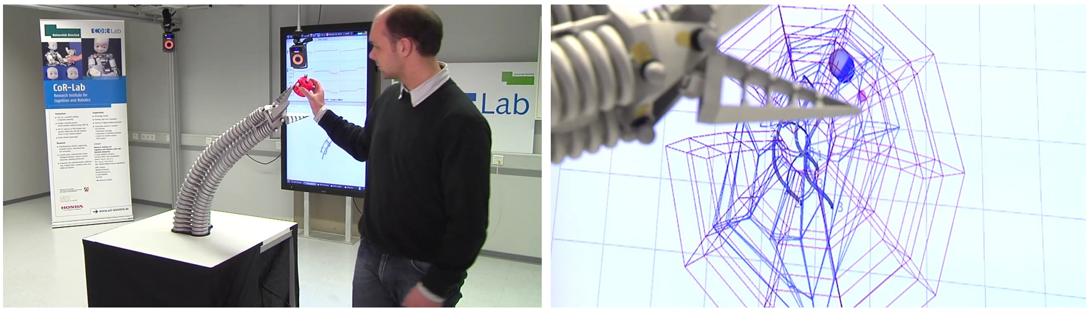

Figure 8. The learned inverse kinematics model can be used to follow moving objects with the end-effector and grasp them (left). During learning and exploration, we use the visualization based on the kinematic simulation extensively to show the robot in relation to its internal, self-generated goals (right). Also, we use the simulation to predict in realtime for the entire space of goals (red) how well the model performs (blue).

5.3 *Use Case 3: End-Effector Control with Goal Babbling*

The BHA is designed to manipulate objects with its end-effector which has three flexible fingers. Unlike other continuum robots it does not possess the mechanical flexibility to wrap around objects with its entire structure in order to lift them, but has to fully rely on a versatile control and usage of its end-effector. Given the length controller, the task reduces to solve the inverse kinematics, i.e. to find the right actuator lengths in order to reach for Cartesian coordinates. One way of solving the inverse kinematics problem is to use the approximate forward kinematics (see Sect. 3.1.2) and invert a local linearization analytically, or to use a numerical approximation of the inverse. However, there occur systematic errors due to the inherent model inaccuracies which can lead to large end-effector deviations in the order of tens of centimeters.

We therefore demonstrated earlier that direct learning the inverse kinematics of the BHA by goal babbling [21] leads already to very accurate approximations. This strategy was inspired from infant developmental studies and mimics how infants attempt early goal-directed movements [30], which structures exploration in a goal directed manner, scales to at least 50 dimensions, and achieves human-competitive learning speed [31]. The approach thereby does not explore and utilize all possible redundancy resolutions or shapes that bring the effector to the same position. It rather explores one consistent redundancy resolution in a highly efficient way which enables the application of the algorithm on high-dimensional morphologies of real-world robots like the BHA. A novel combination of the learned inverse kinematics model with a feedback I-controller further increases the accuracy of the end-effector controller to a remarkable accuracy of a median error below one centimeter [16].

We discuss goal babbling here from point of view of the system architecture, because many different components need to interact in this scenario. First, fast access to the already learned model is crucial, because the exploration is done online and executes and re-adapts the learned model in every single time-step. Second, an efficient goal-directed exploration respecting real-time constraints requires an effective and fast length controller as implemented with the inverse equilibrium model integrated into the central *Control Interface Library*. Finally, the visualization enabled to show the relation between the current robot movement, the goal coordinates, and the robot's workspace which was indispensably helpful during the development of both the learning and the control. We also utilized the high computational efficiency of the forward kinematics approximation to predict in realtime how well the learner performs throughout the entire workspace (see Fig. 8) by means of massive sampling of the inverse model at the goal positions and substituting the approximation for the forward execution of the BHA.

Due to the modularization abilities of the proposed software framework, the learning of the inverse model, the evaluation and a technical demonstration are separated into multiple applications. This allows to learn multiple models and a to switch between them. Once more this use case exploits the flexibility provided by keeping the learned model external and loadable at runtime. Fully integrated, the exploration loop can run online, learn the inverse kinematics module by recruiting most of the described elements of the overall architecture. The learned model can then be reused in other applications, for instance for tracking and grasping (see Fig. 2).

6. Conclusion

Soft robots are a promising approach to human-robot-interaction due to their inherent safety and natural movement behaviors. Potential applications include manufacturing as well as home automation. However, many of the rather experimental platforms that are available today have not been shown to reliably and consistently perform higher level tasks, let alone the problem of how complex the programming of such tasks would be. Nevertheless, advances in understanding and control of compliant actuators strongly support the development of artificial limbs and improve rehabilitation quality [32], [33]. They also pave the way towards support devices like lightweight powered exoskeletons [34], which could ensure the mobility for older people suffering from amyotrophia or support employees in heavy industries.

Implementation of higher level functionalities raises many questions that go beyond basic control strategies and understanding of mechanics, which we tried to address in this paper. Although each component of our system can be considered separately, the composition of modules to complex robot frameworks with advanced functionalities requires an elaborated and structured architecture. We emphasized the benefits of the proposed framework by several examples and are convinced that such a structured working environment is an essential requirement to successfully cope with complex robotic systems in general, and soft robots in particular. Looking back to the requirements and the lack of portable models from rigid body movements, we believe that continuum robots in particular demand flexible software integration to incorporate new concepts, e.g. for compliant interaction, and for learning their internal models.

Soft robots naturally address and implement morphological computation: the mechanics and the bodily physics provide means for embodiment of seemingly sophisticated function. We presented a striking example of this by realizing actively compliant actuator control by utilizing the robot's passive compliance together with a learned model. Here none of the classical complicated mechanisms of impedance control are involved. There is no need for force sensing, no complete inverse dynamics model, no computed torque control, and no impedance regulation. While the accuracy of the presented control mode is limited, it still provides an interesting blueprint for the exploitation of morphological computation. In our scenarios, model learning and hybrid control approaches are able to tackle the problems of inaccuracies caused by the infeasibility of modeling all physical properties of such complex systems and material fatigue of soft materials.

In either case, the full potential of the "softness" is finally leveraged through engineering a control architecture, which relies on powerful tools from software engineering and middleware, together with a mix of classical modeling and machine learning. Our approach for the BHA certainly is not the only possible architecture, but it displays a number of general principles: hardware and software abstractions are important, where useful units may be defined across sensors or actuators, e.g. by decomposing the BHA rather into segments than single actuators; learned modules should be flexibly usable and possibly kept separable from the inner control loops while being heavily employed by them; physical features of the robots may be exploited by re-thinking and approximating classical control in hybrid control schemes; online and real-time learning is a key to cope with non-stationarity. A very important lesson we learned is that exploiting the power of soft robotics, morphological computation, and novel actuation does not mean to dispense with careful control architecture design. Quite the opposite is the case: the demand to integrate novel combinations of control and learning, together with the challenges posed by mechanical properties of the robot rather calls for creative, but well engineered solutions if soft robots shall ever be lifted beyond simple experimentation to perform well defined tasks.

Acknowledgements

J. F. Queißer was funded by the DFG, Cluster of Excellence 277 Cognitive Interaction Technology, R. F. Reinhart and K. Neumann were partially founded by the German BMBF within the Intelligent Technical Systems Leading-Edge Cluster.

References

[1] A. Ijspeert, A. Crespi, and J.-M. Cabelguen, “Simulation and robotics studies of salamander locomotion: applying neurobiological principles to the control of locomotion in robots.” *Neuroinformatics*, vol. 3, no. 3, pp. 171–195, 2005.

[2] M. Schilling, J. Paskarbeit, T. Hoinville, A. Hüffmeier, A. Schneider, J. Schmitz, and H. Cruse, “A hexapod walker using a heterarchical architecture for action selection,” *Front. in Computational Neuroscience*, vol. 7, 2013.

[3] A. Transeth, R. Leine, C. Glocker, K. Pettersen, and P. Liljeback, “Snake robot obstacle-aided locomotion: Modeling, simulations, and experiments,” *IEEE Trans. Robotics*, vol. 24, no. 1, pp. 88–104, 2008.

[4] S. Seok, C. Onal, K.-J. Cho, R. Wood, D. Rus, and S. Kim, “Meshworm: A peristaltic soft robot with antagonistic nickel titanium coil actuators,” *IEEE Trans. Mechatronics*, vol. 18, no. 5, pp. 1485–1497, 2013.

[5] A. Spröwitz, A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri, and A. Ijspeert, “Towards dynamic trot gait locomotion: Design, control, and experiments with cheetah-cub, a compliant quadruped robot,” *Int. Journal of Robotics Research*, vol. 32, no. 8, pp. 932–950, 2013.

[6] H. Hauser, A. J. Ijspeert, R. M. Füchslin, R. Pfeifer, and W. Maass, “Towards a theoretical foundation for morphological computation with compliant bodies,” *Biological Cybernetics*, vol. 105, no. 5-6, pp. 355–370, 2011.

[7] A. Jiang, E. Secco, H. Wurdemann, T. Nanayakkara, P. Dasgupta, and K. Athoefer, “Stiffness-controllable octopus-like robot arm for minimally invasive surgery,” in *Workshop on New Technologies for Computer/Robot Assisted Surgery (CRAS 2013)*, 2005.

[8] M. Calisti, M. Giorelli, G. Levy, B. Mazzolai, B. Hochner, C. Laschi, and P. Dario, “An octopus-bioinspired solution to movement and manipulation for soft robots,” *Bioinspiration & Biomimetics*, vol. 6, no. 3, 2011.

[9] C. D. Marchese, Andrew D andcapable of escape maneuvers using fluidic elastomer actuators,” *Soft Robotics*, vol. 1, no. 1, pp. 75–87, 2014.

[10] K. J. Korane, “Robot imitates nature,” *Machine Design*, vol. 82, no. 18, pp. 68–70, 2010.

[11] A. Grzesiak, R. Becker, and A. Verl, “The Bionic Handling Assistant - A Success Story of Additive Manufacturing,” *Assembly Automation*, vol. 31, no. 4, pp. 329–333, 2011.

[12] N. Kuppuswamy and J.-P. Carbajal, “Learning a curvature dynamic model of an octopus-inspired soft robot arm using flexure sensors,” *The European Future Technologies Conference and Exhibition*, vol. 7, pp. 294 – 296, 2011.

[13] R. F. Shepherd, F. Ilievski, W. Choi, S. A. Morin, A. A. Stokes, A. D. Mazzeo, X. Chen, M. Wang, and G. M. Whitesides, “Multigait soft robot,” *Proceedings of the National Academy of Sciences*, vol. 108, no. 51, pp. 20 400–20 403, 2011.

[14] M. Giorelli, F. Renda, G. Ferri, and C. Laschi, “A feed forward neural network for solving the inverse kinetics of non-constant curvature soft manipulators driven by cables,” in *ASME 2013 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers, 2013.

[15] M. Rolf and J. J. Steil, “Constant curvature continuum kinematics as fast approximate model for the bionic handling assistant,” in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2012, pp. 3440–3446.

[16] M. Rolf and J. Steil, “Efficient exploratory learning of inverse kinematics on a bionic elephant trunk,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 6, pp. 1147–1160, 2014.

[17] 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,” *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, In Press.

[18] A. Nordmann, M. Rolf, and S. Wrede, “Software abstractions for simulation and control of a contin-

uum robot," in *Simulation, Modeling, and Programming for Autonomous Robots.* Springer, 2012, pp. 113–124.

[19] J. Wienke and S. Wrede, "A middleware for collaborative research in experimental robotics," in *IEEE/SICE Int. Symposium on System Integration (SII)*, 2011, pp. 1183–1190.

[20] VICON Motion Tracking Systems. [Online]. Available: http://www.vicon.com

[21] M. Rolf, J. J. Steil, and M. Gienger, "Goal babbling permits direct learning of inverse kinematics," *IEEE Trans. Autonomous Mental Development*, vol. 2, no. 3, pp. 216–229, 2010.

[22] K. Neumann, M. Rolf, and J. J. Steil, "Reliable integration of continuous constraints into extreme learning machines," *Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 21, pp. 35–50, 2013.

[23] Y. Tassa, T. Wu, J. Movellan, and E. Todorov, "Modeling and identification of pneumatic actuators," in *IEEE Int. Conf. Mechatronics and Automation (ICMA)*, 2013.

[24] K. Neumann, M. Rolf, and J. J. Steil, "Reliable integration of continuous constraints into extreme learning machines," *Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 21, pp. 35–50, 2013.

[25] D. Trivedi, A. Lotfi, and C. Rahn, "Geometrically exact models for soft robotic manipulators," *IEEE Trans. on Robotics*, vol. 24, no. 4, pp. 773–780, 2008.

[26] D. Rucker, B. Jones, and R. Webster, "A geometrically exact model for externally loaded concentric-tube continuum robots," *IEEE Trans. on Robotics*, vol. 26, no. 5, pp. 769–780, 2010.

[27] W. Wang, R. N. Loh, and E. Y. Gu, "Passive compliance versus active compliance in robot-based automated assembly systems," *Industrial Robot: An International Journal*, vol. 25, no. 1, pp. 48–57, 1998.

[28] A. Nordmann, C. Emmerich, S. Rüther, A. Lemme, S. Wrede, and J. J. Steil, "Teaching nullspace constraints in physical human-robot interaction using reservoir computing," ser. International Conference on Robotics and Automation, 2012, pp. 1868–1875.

[29] S. Wrede, C. Emmerich, R. Grünberg, A. Nordmann, A. Swadzba, and J. Steil, "A user study on kinesthetic teaching of redundant robots in task and configuration space," *Journal of Human-Robot Interaction*, vol. 2, pp. 56–81, 2013.

[30] C. von Hofsten, "Eye-hand coordination in the newborn," *Developmental Psychology*, vol. 18, no. 3, pp. 450–461, 1982.

[31] M. Rolf, J. J. Steil, and M. Gienger, "Online goal babbling for rapid bootstrapping of inverse models in high dimensions," in *IEEE Int. Joint Conf. Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, 2011.

[32] H. Yu, S. Huang, N. V. Thakor, G. Chen, S.-L. Toh, M. Sta Cruz, Y. Ghorbel, and C. Zhu, "A novel compact compliant actuator design for rehabilitation robots," *IEEE Int. Conf. on Rehabilitation Robotics*, pp. 1–6, 2013.

[33] Y. S. Song, Y. Sun, R. van den Brand, J. von Zitzewitz, S. Micera, G. Courtine, and J. Paik, "Soft robot for gait rehabilitation of spinalized rodents," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2013, pp. 971–976.

[34] M. Wehner, B. Quinlivan, P. M. Aubin, E. Martinez-Villalpando, M. Baumann, L. Stirling, K. Holt, R. J. Wood, and C. J. Walsh, "A lightweight soft exosuit for gait assistance." *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pp. 3362–3369, 2013.