# Probabilistic Automata Model of a Soft Robot for the Planning of Manipulation Tasks

Martin Stommel, Zhicong Deng, and Weiliang L. Xu

*Abstract*—**Soft robots must be able to structure an automation problem into a sequence of actions that lead to a desired state, before they can fulfill a meaningful role in automation applications. This, however, can only be successful if the robot can predict the outcome of an action. The theory of rigid industrial robots is not applicable without major changes, because kinematic chains do not adequately describe the continuous deformation of the complex, often biologically inspired shapes of soft robots. Analytic solutions have not been found yet. Numerical solutions based on finite elements are slow, technically challenging, and only suitable for one specific robot. It is, however, possible to observe the outcome of an action, and use these observations to plan a sequence of actions that let the robot accomplish an automation task. In this paper, we analyze a probabilistic automaton that computes the optimal sequence of actions to bring the robot into a desired state. An earlier article explained the functioning of the method in a toy example. In this paper, we analyze if it is feasible to apply the method to a planning problem inspired by a real soft robot. We show the results and document the planning process. We identify the analog of an impulse response, although it is not closed form due to the nonparametric nature of the method.**

*Note to Practitioners*—**A soft robotic sorting table has a computer-controlled soft surface that can move delicate objects without damaging them. There are currently no closed-loop control systems for such robots, because it is unclear how to relate the control signals to the behavior of the table, or which actions to choose in order to solve a manipulation task. In this paper, we propose a probabilistic automaton to plan the best action sequence on average. The sequence brings the workpieces on top of a soft table into a desired condition. It is a machine learning solution that is based on observations of the input signals and their effect, rather than a detailed analytical or numerical modeling of the robot. We show that it is feasible to model an existing soft robotic table. We demonstrate that the planning is successful by solving complex maze tasks. Our results are based on experiments and simulations.**

*Index Terms*—**Control, planning, soft robotics.**

## I. Introduction

SOFT robots are biologically inspired robots that are built to realize some of the advantages of soft natural tissue in an engineering design. These advantages can be new capabilities or suitabilities for new applications. The most important property is that a soft robot naturally adapts to the surroundings without need for complex control systems. This is an advantage for industrial applications, where delicate objects may not be damaged [1] or there are safe human–robot interactions and human–motor assistance [2]. From a computational point of view, the interaction of the soft robot body with its environment can be considered as a morphological computation, which would be very difficult to establish by analytic approaches [3], [4]. Soft robotic simulators of the human esophagus and stomach can be used in medical and food research [5]–[7]. Walking and crawling robots are used to investigate locomotion patterns [8], [9]. Soft grippers and tentacles have been developed to handle fragile objects [10]–[12]. The prospect of adapting the shape and locomotion strategy to challenging terrain makes soft robots interesting for military reconnaissance, natural disaster relief, and pipe inspection [2].

In this paper, we consider mainly applications in industrial automation where the flexibility of a soft robot would be welcoming (e.g., in human–robot interaction) although, generally, the environment would have been set up to suit the task. The control algorithm would be required to perform a flexible sequence of actions (e.g., to prepare a workpiece for some critical process step) depending on the sensed current situation. At the same time, we assume that the environment is sufficiently structured, so that there is a limit to the number of situations that the robot can encounter.

To control a robot in an automation task, the control signals must be related to the behavior of the robot and the workpiece. This has been achieved only partially in soft robotics. Most research focuses on the control of single mechanical actuators to realize a simple movement. Wang and Iida [13] distinguish the movement types elongation/shortening, bending, flowing, and reconfiguration. The aim of these studies is to prove the correct functioning of a new actuator or to demonstrate a new robotic ability [14], often in comparison to a biological example [15], [16]. The next higher level is to repeat simple movements. This is usually done to realize repetitive locomotion patters in crawling or walking robots [9], [17]. The behavior of the robot is typically not linked to the solution of a certain task, although closed-loop control can be used to regulate the pressure of a pneumatic actuator or to control the degree of bending over a part of the soft structure.

To control the behavior of a soft robot with respect to an automation task, it is necessary as follows:

1) to have sensor feedback with respect to the task;
2) to have a model that links the control signals to the movement of the robot;

Manuscript received December 14, 2016; revised April 23, 2017; accepted June 7, 2017. Date of publication July 3, 2017; date of current version October 4, 2017. This paper was recommended for publication by Associate Editor A. Pashkevich and Editor H. Ding upon evaluation of the reviewers' comments. *(Corresponding author: Martin Stommel.)*

M. Stommel is with the Department of Electrical and Electronic Engineering, Auckland University of Technology, Auckland 1010, New Zealand (e-mail: mstommel@aut.ac.nz).

Z. Deng and W. L. Xu are with the Department of Mechanical Engineering, University of Auckland, Auckland 1010, New Zealand (e-mail: p.xu@auckland.ac.nz).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2017.2714167

3) and to have a model that links the movements of the robot to the task.

A variety of sensors have been proposed for soft robotics [18], [19]. Although this is an open and interesting topic, a discussion on sensor technologies is not in the scope of this paper. Here, we assume that a sensor technology exists that provides a classification of the current situation.

Modeling is often restricted to predicting the parameters of a single actuator or the bending of a thin elongated structure [20]. More complex robot structures have been discretized in a way that allowed for the transfer of modeling techniques known from rigid robotics and continuum mechanics. Duriez [21] proposes a finite elements method to simulate the shape of a soft robot in real-time. Largillière *et al.* [22] extend the method to compute the mechanical forces required to achieve a certain shape. A soft, continuously bending arm has been modeled approximately by a kinematic chain over a high number of discrete points [23], [24]. Renda *et al.* [25], [26] developed a piecewise constant strain model as a discretized approximation of the Cosserat beam. The shape of a robot is determined numerically. The models are usually tailored toward a particular robot. Also, knowledge of the deformation of a robot does not imply knowledge of the interaction with a workpiece or the actions required to solve a multistep automation problem.

Machine learning has been proposed to bridge the gap between control signals and the progress of an automation task. Because of the exponential complexity of coupled actuators in a joint robot body, machine learning is unable to model all degrees of freedom. However, for a restricted number of behaviors of interest, the control signals can be optimized to realize the required robot shapes [27]. A probabilistic automaton has been proposed to link a set of robot actions to the solution of an automation task [28]. The automaton was able to predict the system behavior based on observations of the state transitions triggered by performing certain actions. A cost function was proposed to guide the selection of the most promising action to reach a desired state. The approach was shown to be computationally efficient, i.e., of polynomial complexity. But there was no experimental or other evidence that it is feasible to create such an automaton in practice.

The aim of this paper is to provide support for the following hypotheses.

1) It is realistic to create the state-transition matrix.
2) The cost function allows to predict the number of actions required to achieve a desired state.
3) It is realistic to plan the action sequence.
4) The cost function allows for the planning of robot actions to achieve a desired state.

The term "realistic" is used to express that a suitable solution can be found in acceptable time on consumer-level hardware. This is an important criterion for the practitioner that differs from the asymptotic complexity in the limit of large inputs. However, the experimental setting described in this paper is optimized toward comprehensiveness and ease of illustration rather than the solution of a commercially relevant problem.

## II. Automatic Structuring of the Automation Task

The (continuous) system state is a continuous time signal that consists of the values of all parameters (control signals and sensor measurements) that are relevant for the solution of an automation task. For computational convenience, we discretize both the system state space and time. Each partition of the discretized state space is assigned a unique natural number, the state number $s \in S \subset \mathbb{N}$, or state $s$ for short. Let ${}^{n}s$ denotes the state at discrete time $n$. We assume that ${}^{n}s$ can be obtained by a suitable sensor technology. The goal of the automation task is to bring the system from an initial state ${}^{0}s \in S$ to a desired final state $f \in F \subset S$. Repetitive or more complex automation tasks can be solved by choosing a new final state after $f$ has been reached.

Let $a \in A$ denotes the action performed by the soft robot during one step in time. The set $A$ consists of all movement primitives that a soft robot can produce. We do not make many assumptions about the action itself, but note that many studies have demonstrated the effective operation of soft robotic actuators. A probabilistic automaton $(S, A, \delta, {}^{0}\mathbf{s}, F)$ will be used to select those actions that lead to the solution of the automation task [28]. The probabilistic formulation accounts for shortcomings (mostly noise) in the measurement of the system state.

The functionality of the robot is encoded in the probabilistic state transitions $\delta : S \times A \times S \mapsto [0\ 1]$, where

$$\delta(s_k, a, s_j) = p(s_j | s_k, a) = [M_a]_{s_j s_k} \tag{1}$$

describes the effect of performing action $a$ in state $s_k$ by probability distributions over successor states. They can be stored in the state-transition matrices

$$M_a = \begin{bmatrix} & & p_{a0k} & \\ & & \vdots & \\ p_{aj0} & \cdots & p_{ajk} & \cdots \\ & & \vdots & \end{bmatrix} \tag{2}$$

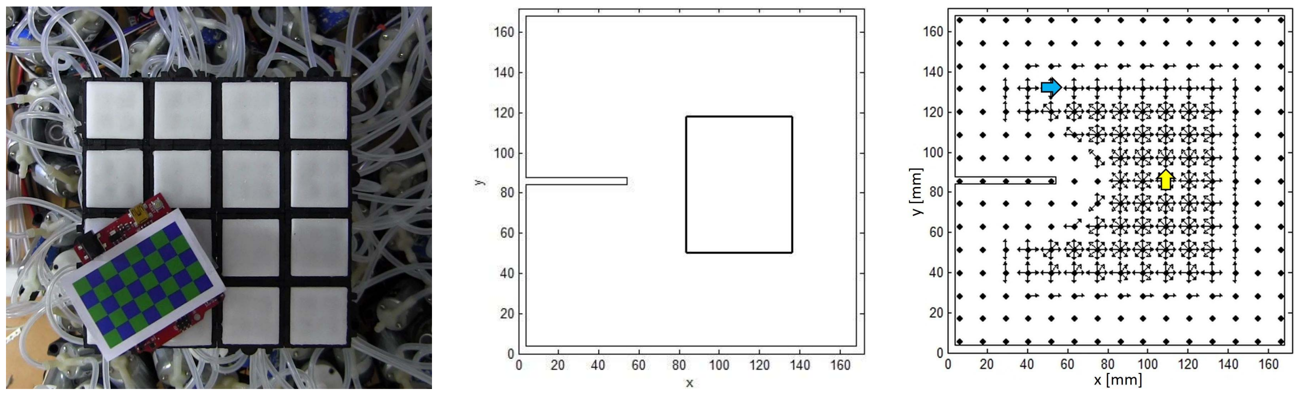
where $p_{ajk} = p(s_j | s_k, a)$ is the probability of moving from state $s_k$ to $s_j$ by performing action $a$. The matrix is a description of the probable development of the system state from time $n$, when ${}^{n}s = s_k$, to time $n+1$, when ${}^{n+1}s = s_j$. However, neither $\delta$ nor $M_a$ depends on the time $n$. Since $M_a$ gives distributions over successor states, each column of the matrix integrates to one. Hypothesis (1) states that it is possible to find this matrix for a real robot.

The possible sequences of actions that the robot can take to reach a final state differ by their length $l$ and the probability of realizing the expected state changes.

The probability $p_1(a, s)$ of reaching any final state in a single step by performing action $a$ can be computed by tracing back the state-transition probabilities from the final states to the current state $s$ by computing

$$p_1(a, s) = \sum_{f \in F} p(f | s, a). \tag{3}$$

For longer sequences, there are usually multiple sequences that start with the same first action $a$ in state $s$. For any



(a) Soft table (b) Simulated scenario (c) Simulated workspace

Fig. 1. Simulated scenario. (a) Top view of a soft robotic table. A printed circuit board serves as workpiece in our experiments. It is marked by a checkerboard pattern to support visual tracking. The background shows the air pumps, solenoid valves, and rotary pumps of the pneumatic actuators. (b) Geometry of the simulated soft table with obstacles around the border and one wall on the left that extends toward the center. The rectangular black dots mark the size of the workpiece at a region where the workpiece is able to rotate without collision with the obstacles. (c) Discrete state space of the robot and workpiece. Small black arrows indicate the workpiece orientation in physically plausible configurations.

fixed length of a sequence, we always choose the action that yields the highest probability of reaching a final state via those sequences. The probability of reaching the final state in maximally $L$ steps is then given by

$$\tilde{p}_L(a,s) = \sum_{\tilde{s}\in S} p(\tilde{s}|s,a) \max_{\tilde{a}\in A} \tilde{p}_{L-1}(\tilde{a},\tilde{s}), \quad 1 < L \tag{4}$$

and

$$\tilde{p}_1(a,s) = p_1(a,s). \tag{5}$$

The tilde is used as text decoration. By comparing sequences of different length, we obtain the probability

$$p_l(a,s) = \tilde{p}_l(a,s) - \tilde{p}_{l-1}(a,s), \quad 1 < l \tag{6}$$

of reaching the final state in exactly $l$ steps. Hypothesis (2) refers to $p_l(a,s)$.

A cost function is introduced to minimize the time that the robot spends in error states $s \notin F$. We define the cost $c$ as the average number of steps required to bring the robot from state $s$ to any final state $f$ by performing action $a$ next, as well as further actions if needed. In practice, it might be useful to assign an additional penalty to certain states, for example, if they require frequent manual or other intervention. The cost can be computed by weighting the length of a sequence by the probability of realizing the required state transitions according to

$$c(s,a) = \sum_{l=1}^{L} l \, p_l(a,s). \tag{7}$$

The best action

$$a_{\text{opt}} = \arg \max_{a\in A} c(s,a) \tag{8}$$

to perform at any state $s$ is the one that minimizes the cost. Hypothesis (3) states that we can compute (8). Hypothesis (4) states that the result is correct.

## III. Modeling of a Soft Robotic Sorting Table

A simulation of an existing soft robot is conducted to analyze the feasibility of the approach. A maze task is chosen for the simulation, because it is complex enough to be interesting, but at the same time simple enough to be documented in an illustrative and instructive way. After defining the task, we construct the state-transition table from measurements of a real soft robotic sorting table. Then, we use these measurements to let a probabilistic automaton solve the planning task.

### A. Simulated Scenario

By design, the soft robotic sorting table [Fig. 1(a)] is capable of shifting and rotating objects on the surface by applying certain patterns of control signals to the pneumatic actuators over multiple cycles [29].

The simulated setting consists of a soft robot with a surface of $172 \times 172$ mm$^2$, a flat workpiece of $68 \times 53$ mm$^2$, static obstacles on the robot surface, and a top-mounted camera that provides the position and orientation of the workpiece [Fig. 1(b)]. The existing prototype of the soft table consists of a set of actuator modules that can be configured to a $4 \times 4$- or $5 \times 5$-array with an area of $144 \times 144$ mm$^2$ and $180 \times 180$ mm$^2$, respectively. Because of the size of the workpiece, actuation near the border is successful as well, so both configurations match the simulated scenario. We define a coordinate system with the axes $x$ and $y$ for a position on the surface, and $\theta$ for the angle of the workpiece. The dimensions of the workpiece are taken from a printed circuit board, which we used in the practical experiments.

The obstacle consists of a 4-mm wide border and a 50-mm bar on the left that points toward the center. It is representative of the challenges found in an industrial application, where some regions of the state space are not accessible due to the specific tools and processes. The obstacle exists only in the simulation but not in the physical prototype. Because of the obstacle, there is no enough space on the left

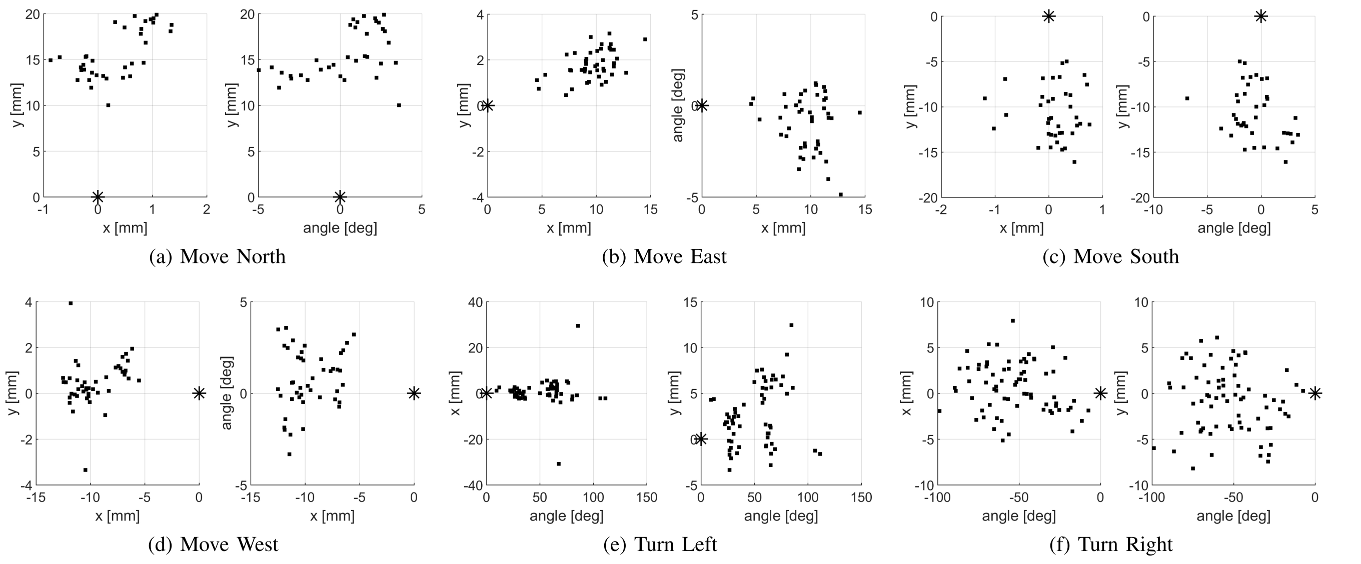


Fig. 2. Observed robot behavior: each subplot shows the effect of a certain action measured over multiple experiments. The initial workpiece position is marked by the star symbol. The scatter plots show the distribution of workpiece positions and orientations relative to the initial position. The initial position is set to $(0, 0, 0)^\top$. In the experiments, however, the initial positions were spread out across the surface of the soft table. (a) Move north. (b) Move east. (c) Move south. (d) Move west. (e) Turn left. (f) Turn right.

side for the workpiece to rotate. It fits on either side of the obstacle, but cannot turn. On the right side of the robot surface, there is enough space for the workpiece to rotate freely. This creates a sufficiently complex scenario for the simulation of an automation task, because moving the workpiece through the obstacles usually requires a sequence of different movements depending on the start position and orientation.

We model six different actions based on the inverted caterpillar movement proposed for the robot [29]. Each action shifts or rotates the workpiece for a number of cycles.

1) Move north: translation along the $y$-axis for 7 cycles.
2) Move east: translation along the $x$-axis for 7 cycles.
3) Move south: translation against the $y$-axis for 7 cycles.
4) Move west: translation against the $x$-axis for 7 cycles.
5) Turn left: counterclockwise rotation for 15 cycles.
6) Turn right: clockwise rotation for 15 cycles.

On average, these actions move the workpiece by 11 mm in one direction, or turn it by 52°.

The aim of the simulation is to plan the sequence of actions required to move the workpiece to a desired coordinate (position and angle) for any start position.

## *B. Modeling of the States and State Transitions*

The (continuous) state space consists of the variables $x$, $y$, and $\theta$. It is discretized into 15 steps each for $x$ and $y$, and eight steps for the angle $\theta$. This results in $15 \times 15 \times 8 = 1800$ distinguishable states. Because of collisions with the obstacles, not all states are physically reachable. Fig. 1(c) shows the workspace considered in our simulation: rectangular black dots show the center points of the partitioning along the $x$ and $y$ coordinate. Small black arrows indicate the orientation of the workpiece for a reachable state. Some states are only reachable for a small subset of the corresponding continuous values. No arrow means that the state was unreachable. The colored arrows indicate the start state (yellow) and final state (blue) chosen in one of the experiments presented later.

The effect of performing the actions was determined experimentally using the real robot and supported by a physical model of the workpiece–obstacle interaction. We first calibrated the top-mounted camera using the MATLAB calibration toolbox. This reduces errors by lens distortion and allows for a transform from pixel coordinates to world coordinates with submillimeter accuracy. We recorded 111 videos of workpiece movements on different positions of the table. The position was tracked by using an optical marker that allowed for the automatic computation of the parameters $x$, $y$, and $\theta$. This resulted in 35 to 72 observations of the initial and posterior position/orientation per action (Fig. 2), which was sufficient to model the effect of each action $a$ by a $3 \times 3$ covariance matrix $\Sigma_a$ of $x$, $y$, and $\theta$. The standard deviation was 2.4 for the translational movements and 55 for rotations (measurements in mm and degrees). The distribution of posterior positions is not symmetrical to the axes because the workpiece itself is not symmetrical.

To reduce the number of experiments with the physical robot, we used a simple physical model to predict the behavior in the case that the workpiece collides with an obstacle. Accurate physics modeling is not in the scope of this paper, so we used only a very basic model. The model prevented the workpiece from traveling through obstacles and allowed for the rotation around a point of contact, as well as the alignment of the straight edges of the workpiece with the obstacles. In a real application, a more sophisticated physical model would be required. To combine the stochastic and physical model, we defined the initial position $({}^0x, {}^0y, {}^0\theta)^\top$ of the workpiece and sampled the posterior position $({}^1x, {}^1y, {}^1\theta)^\top$ from a multivariate normal distribution

$$({}^1x, {}^1y, {}^1\theta)^\top - ({}^0x, {}^0y, {}^0\theta)^\top \sim \mathcal{N}_3(\boldsymbol{\mu}_a, \Sigma_a) \qquad (9)$$

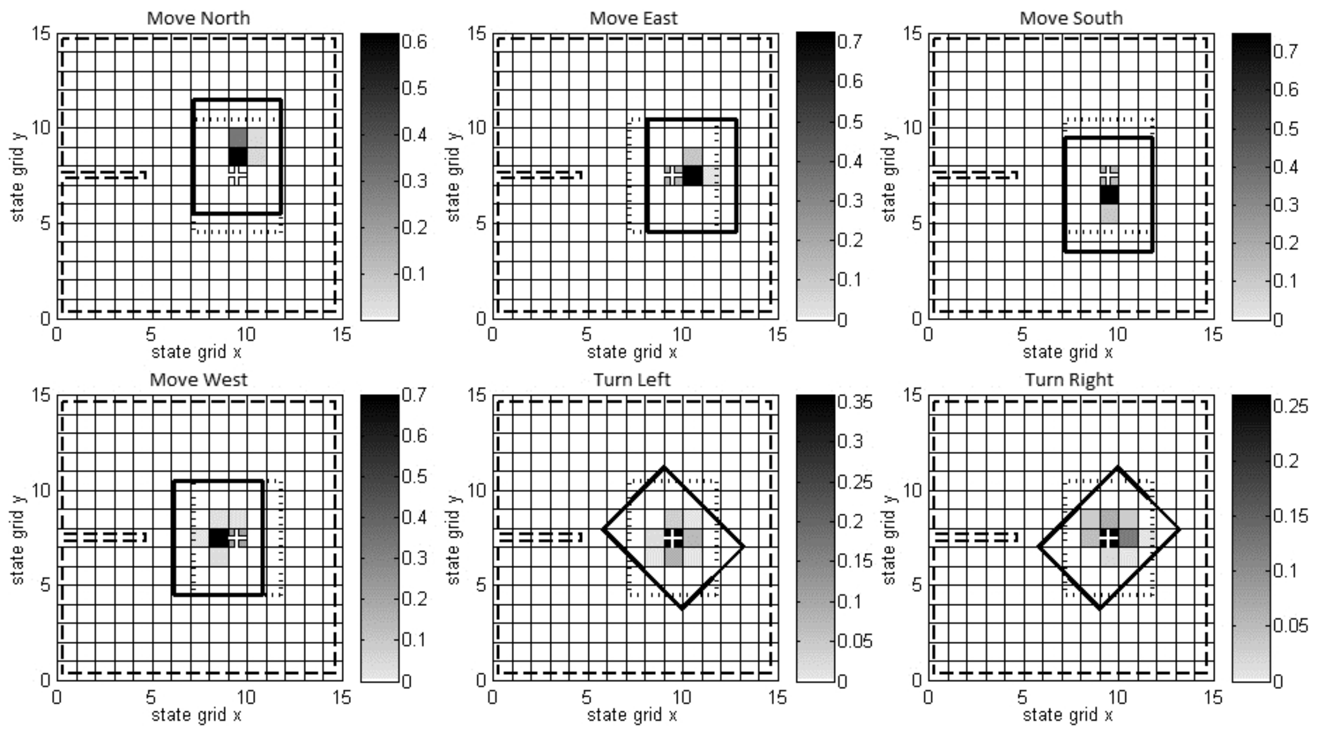
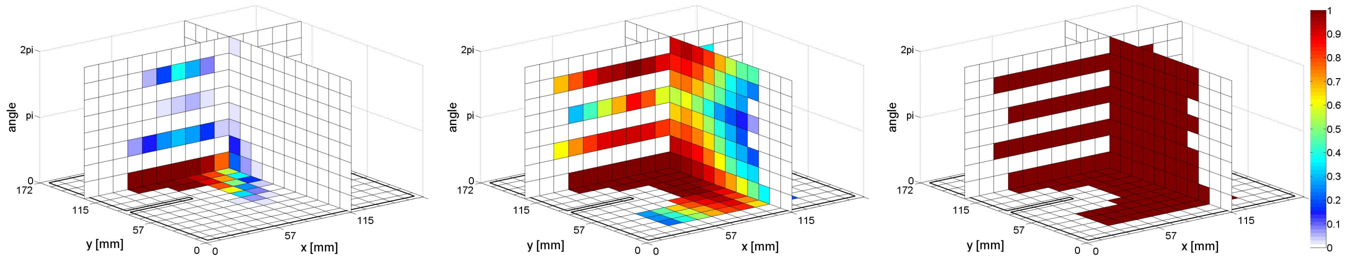


Fig. 3. Planar projection ($x, y, \theta$ to $x, y$) of the state-transition probabilities for an initial workpiece position on the mid-right, where the objected can move freely without colliding with an obstacle. Each plot shows one action. The white plus-marker and the dotted rectangle show the initial state and workpiece outline. The bold, solid rectangle indicates the most likely successor state of the workpiece. The probability density of reaching the neighboring positions are color coded.



(a) Max. path length = 6 (b) Max. path length = 12 (c) Max. path length = 25

Fig. 4. Visualization of the planning progress: the algorithm computes the probability $\tilde{p}_L(a, s)$ of reaching a final state for increasingly longer sequences of actions. $\tilde{p}_L(a, s)$ for the value of $a$ maximizes $p$. The state number $s$ is shown in terms of the underlying discretized state variables $x$, $y$, and $\theta$. For clarity, only the results along three selected planes through the 3-D state space are shown. A single final state was chosen as illustrated in Fig. 1(c). For very short sequences of actions, the final state can only be reached if the initial position of the workspace is close to it. For long sequences, the final state can be reached from all considered initial states with high probability (states that were not physically reachable were excluded). The comb-like structure in (c) indicates that the rotation of a workpiece in the upper left area of the workpiece is restricted by obstacles. In order to turn from horizontal to vertical, the workpiece must first move to the central right area, where it can rotate freely. (a) Maximum path length = 6. (b) Maximum path length = 12. (c) Maximum path length = 25.

parameterized by the respective mean $\boldsymbol{\mu}_a$ and covariance matrix. The physical model was applied when intermediate positions caused collisions between the workpiece and the obstacle. The initial position

$$({}^0x, {}^0y, {}^0\theta)^\top \sim \mathcal{U}((0, 0, 0)^\top, (172, 172, 2\pi)^\top) \tag{10}$$

was sampled from the continuous state space along a regular grid with a spacing of 3 mm for $x$ and $y$, and 14.4° for $\theta$. Every action was sampled four times. In an initial step, a number of 1328 states out of 1800 were found unreachable and excluded from the sampling. For the remaining 472 states, this resulted in a total of 2 099 136 simulated state transitions, which were used to compute the state-transition matrices $M_a$. Although this is a relatively dense sampling of close to 194 samples per initial state and action, for a small number of state transitions no observation was obtained. For simplicity, we excluded the respective states from the simulation, resulting in the workspace as shown in Fig. 1(c). In practice, it would be necessary to develop a rule for this, e.g., a nearest neighbor assignment to a similar state with sufficient observations.

## IV. Results

By counting the initial and posterior states, we were able to establish the state-transition matrices from (2). Fig. 3 shows a plot of the state-transition probabilities for a selected initial state [see yellow arrow in Fig. 1(c)] and the six actions.

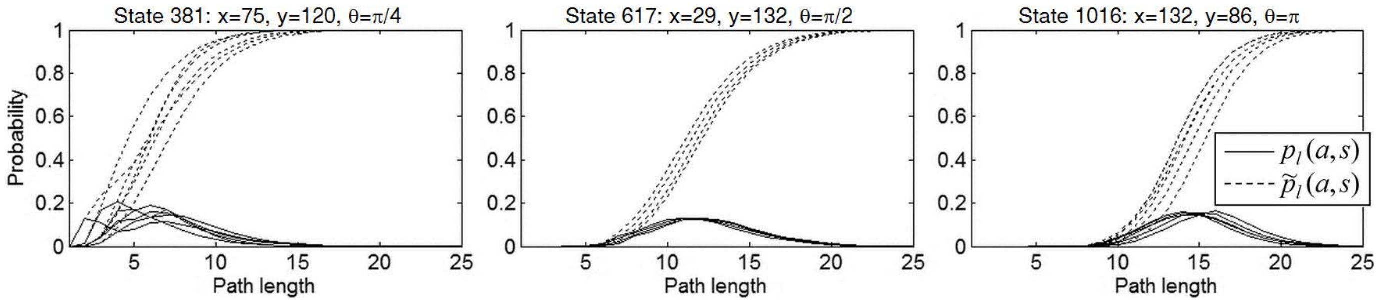


Fig. 5. Probabilities $p_l(a, s)$ and $\tilde{p}_l(a, s)$ of reaching the final state by performing an exact ($p$) or maximum ($\tilde{p}$) number of actions. Results are shown for three randomly selected states $s = 381, 617, 1016$. The corresponding approximate values of $x$,$y$, and $\theta$ are given in the diagram title. The six (dashed and solid) curves represent the six possible actions. Since the state transitions contain a probabilistic element, choosing one action at a particular current state does not always result in the same sequence of states toward a final state. Therefore, we always perform the action that yields the minimum number of state transitions on average.

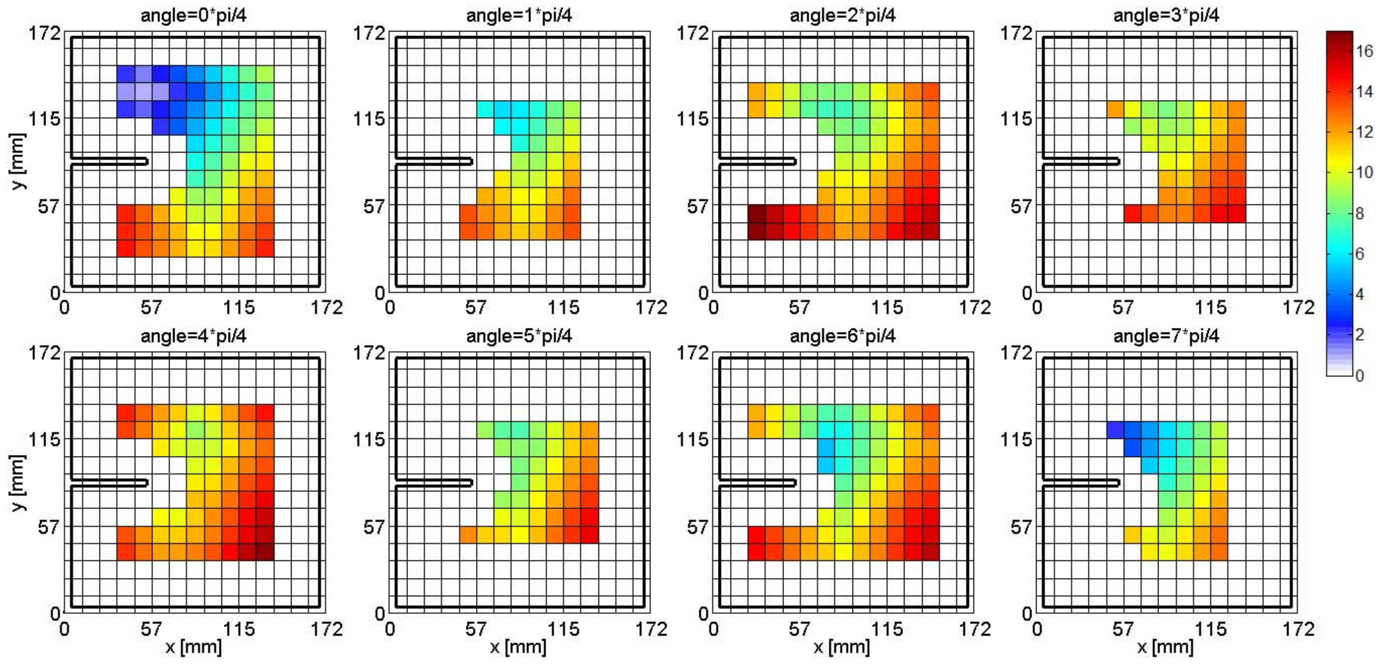


Fig. 6. Expected number of actions $c(s, a)$ to reach a final state. The plots show $c$ for all states $s$. For the illustration, the 3-D state space (consisting of the discretized values of $x$, $y$, $\theta$) has been split up into eight diagrams, each one representing one orientation $\theta$. The horizontal and vertical axes represent $x$ and $y$. The action $a$ is selected to minimize $c$. In this paper, we did not consider sequences of length zero, so one step is reserved to stop the simulation. The plot shows that the cost increases with the distance from the destination coordinate in the upper left area. The cost also increases if the workpiece needs to be reoriented (turning the workpiece by 45° corresponds to moving from one diagram to another). Note that not all orientations are physically possible in all positions, because the workpiece would collide with an obstacle.

The CPU-time of sampling the state-transition probabilities was 2:48 h on an office computer (Intel Core i5 quad-core CPU clocked at 3.2 GHz). This time is acceptable, because establishing the state-transition matrices is an offline step that needs to be done only once for every change in the physical setup of the robot.

The state-transition probabilities allowed us to compute the cost of the best sequences of actions to reach a final state from (almost) all initial states ("almost" means that a final state cannot be reached if the physical setup does not allow for it). Fig. 4 shows the value of $\tilde{p}_L(a, s)$ for $L = 6, 12$, and 25. The final state is chosen as shown in Fig. 1(c). For small values of $L$, the probability of reaching the final state is zero for most actions and states, because the final state is physically out of reach: the number of actions $L$ is too small to move the workpiece over the full distance to a final state. By increasing $L$, the region of high probability of reaching the goal spreads out over the parameter space of $a$ and $s$ starting at the final states. At $L = 25$, the probability of reaching a final state was higher than 99.3% for all valid values of $a$ and $s$.

Fig. 5 shows $p_l(a, s)$ and $\tilde{p}_l(a, s)$ for three fixed values of $s$ but varying values of $l$ and $a$. Due to the probabilistic modeling of the state transitions, the probability of reaching a final state is distributed over different lengths of the sequence. For efficient robot control, we chose the action that yields a minimal length on average. Fig. 6 plots the resulting average length (7) of the sequence for every state. The action that yields the best result (8) is indicated in Fig. 7.

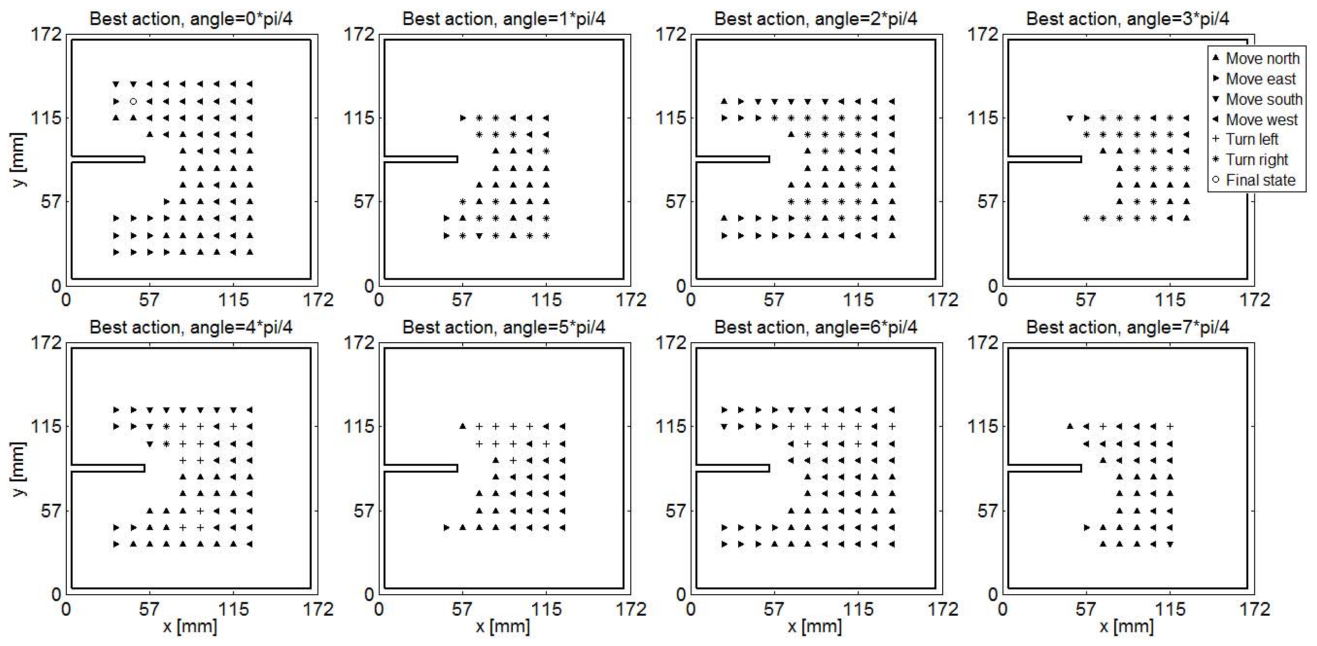


Fig. 7. Best action to reach a final state in minimum time. Every diagram represents one of eight workpiece orientations. The best action is plotted for each workpiece position modeled by a state of the probabilistic automaton.

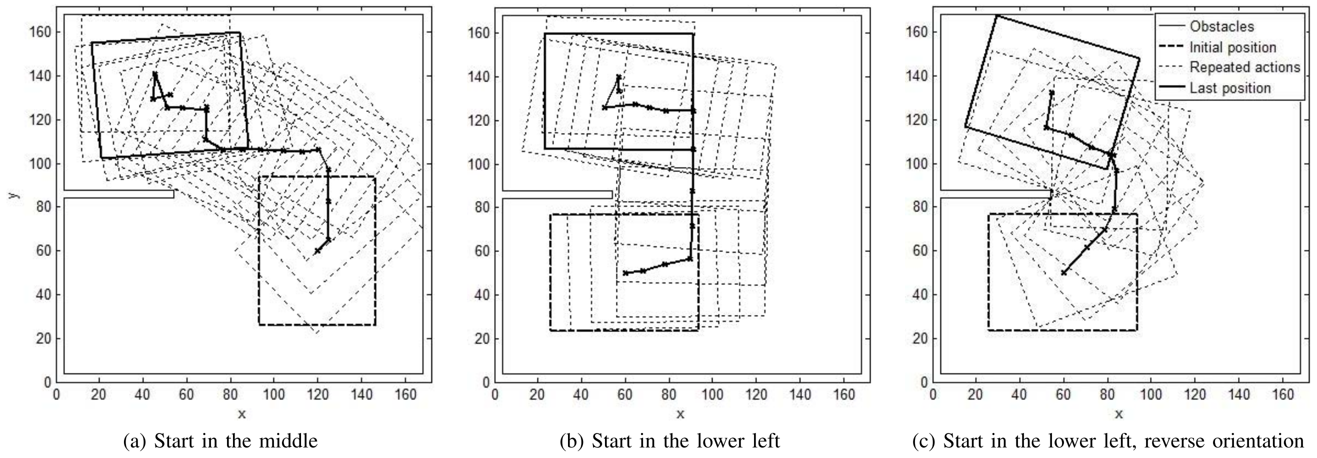


Fig. 8. Example sequences of state changes to reach the final state. The algorithm always selects the best action as identified in the planning stage. The workpiece position after performing an action is sampled from the (known) distribution of posterior positions. (a) Start in the middle. (b) Start in the lower left. (c) Start in the lower left, reverse orientation.

The system can be understood as a control system, where a final state is the set point and all other states correspond to an error that is to be minimized. Since the probabilistic automaton does not require a metric on the states, it is hard to quantify an impulse or the impulse response. However, the values of $p_l(a, s)$ show that the response of the system to an error depends on the error state, and that it is a distribution over the number of actions required to minimize the error. The distribution is computed during the planning of the actions and can be stored in a lookup table for further computations.

The CPU-time to compute the probabilities $p_l(a, s)$ and $\tilde{p}_l(a, s)$, the cost function and the best action was 1.96 s. This is acceptable, since the planning is an offline step that needs to be done only once for a change of the physical setup or a change of the desired final states. Since the cost function and the best actions can be stored in a lookup table, it is not necessary to repeat the planning steps in a repetitive automation task or when switching between multiple tasks.

For further validation, we implemented a probabilistic automaton where we always applied the best action as given by the cost function, and where the successor state was sampled from the state-transition tables. A single final state was defined, as shown in Fig. 1. We did not measure the speed of this simulation, because the best action can be determined efficiently from a precomputed lookup table. Fig. 8 shows example results for three different initial workpiece configurations. In example (a), the workpiece first rotates right and then moves backward into the space on the upper left. In example (b), the workpiece is shifted around to the obstacle to reach the final position. In example (c), the initial orientation is turned by 180° compared with example (b). The robot exploits the interaction with the obstacle to turn the workpiece and move it into its final position.

## V. Conclusion

A previous study suggested that a probabilistic automaton can be used to control a soft robot to solve an automation task. The result was that the algorithms have polynomial complexity, which is commonly considered as a necessary condition for practical relevance. In this paper, we demonstrated that the required number of states, state transitions, and actions is small enough to practically compute the solution of an automation task inspired by the capabilities of an existing soft robot. This is an important step toward the application of soft robots in automation, because it means that it is realistic to model the state space and operation of the robot in the proposed way.

First, we showed that it is possible to obtain the state-transition matrix for a real robot. The behavior of a free moving workpiece was measured directly from the robot. To model the workpiece–obstacle interaction, it was more practical to use a computational model. Since our model was very basic, the state-transition matrix that we obtained allows us to model the behavior qualitatively, but not at high precision. However, better physical models do exist and have been demonstrated in other applications. The computation time of 2:48 h was acceptable for an offline step.

Then, we computed a cost function that allowed us to determine the best action in every state to bring the system into a desired state. The computation also provides the average number of actions to reach a final state, as well as the distribution over the length of successful sequences up to a reasonable maximum length. We showed examples of using the planning result to bring the automaton into a final state. The computation time of the planning was less than 2 s, which is acceptable for an offline step.

We showed that the planned actions are able to move the workpiece to the desired position and orientation. Since the plan can be stored in a lookup table, the proposed method is much faster than any method that contains a numerical computation of the exact robot shape. Our results show that the primary challenges of the method lie in the modeling of the state-transition matrix, but we also showed that a practical solution can be found. We can expect additional challenges from an application of the system to the real-time control of a physical robot.

## Acknowledgment

The authors would like to thank F. Le Daheron, Polytech Nantes, Graduate School of Engineering, University of Nantes, Nantes, France, and A. Le Page, École Nationale Supérieure de Mécanique et d'Aérotechnique, Chasseneuil-du-Poitou, France, for their support in recording and measuring the workpiece motion.

## References

[1] M. Stommel, P. Xu, P. P. K. Lim, and B. Kadmiry, "Robotic sorting of ovine offal: Discussion of a soft peristaltic approach," *Soft Robot.*, vol. 1, no. 4, pp. 246–254, 2014.

[2] C. Majidi, "Soft robotics: A perspective—current trends and prospects for the future," *Soft Robot.*, vol. 1, no. 1, pp. 5–11, Jul. 2013.

[3] C. Paul, "Investigation of morphology and control in biped locomotion," Ph.D. dissertation, Dept. Comput. Sci., Univ. Zurich, Zürich, Switzerland, 2004.

[4] C. Paul, "Morphological computation: A basis for the analysis of morphology and control requirements," *Robot. Auto. Syst.*, vol. 54, no. 8, pp. 619–630, 2006.

[5] S. Dirven, W. Xu, J. Allen, L. K. Cheng, and J. Bronlund, "Biologically-inspired swallowing robot for investigation of texture modified foods," *Int. J. Biomechatron. Biomed. Robot.*, vol. 2, nos. 2–4, pp. 163–171, 2013.

[6] Y. Dang, L. Cheng, M. Stommel, and W. Xu, "Technical requirements and conceptualization of a soft pneumatic actuator inspired by human gastric motility," in *Proc. Int. Conf. Mechatron. Mach. Vis. Pract. (MVIP)*, Nov. 2016, pp. 1–6.

[7] R. Hashem, W. Xu, M. Stommel, and L. Cheng, "Conceptualisation and specification of a biologically-inspired, soft-bodied gastric robot," in *Proc. Int. Conf. Mechatron. Mach. Vis. Pract. (MVIP)*, 2016, pp. 1–6.

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

[9] M. Wehner *et al.*, "An integrated design and fabrication strategy for entirely soft, autonomous robots," *Nature*, vol. 536, pp. 451–455, Aug. 2016.

[10] E. Brown *et al.*, "Universal robotic gripper based on the jamming of granular material," *Proc. Nat. Acad. Sci. USA*, vol. 107, no. 44, pp. 18809–18814, 2010.

[11] F. Ilievski, A. D. Mazzeo, R. F. Shepherd, X. Chen, and G. M. Whitesides, "Soft robotics for chemists," *Angew. Chem. Int. Ed.*, vol. 50, no. 8, pp. 1890–1895, 2011.

[12] C. Laschi, M. Cianchetti, B. Mazzolai, L. Margheri, M. Follador, and P. Dario, "Soft robot arm inspired by the octopus," *Adv. Robot.*, vol. 26, no. 7, pp. 709–727, 2012.

[13] L. Wang and F. Iida, "Deformation in soft-matter robotics: A categorization and quantitative characterization," *IEEE Robot. Autom. Mag.*, vol. 22, no. 3, pp. 125–139, Sep. 2015.

[14] C. Laschi, B. Mazzolai, and M. Cianchetti, "Soft robotics: Technologies and systems pushing the boundaries of robot abilities," *Sci. Robot.*, vol. 1, no. 1, p. eaah3690, 2016.

[15] M. Cianchetti, M. Follador, B. Mazzolai, P. Dario, and C. Laschi, "Design and development of a soft robotic octopus arm exploiting embodied intelligence," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2012, pp. 5271–5276.

[16] G. Runge, S. Zellmer, T. Preller, G. Garnweitner, and A. Raatz, "Actuation principles for the bioinspired soft robotic manipulator spineman," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2015, pp. 1329–1336.

[17] T. S. Zhang, A. Kim, M. Ochoa, and B. Ziaie, "Controllable 'somersault' magnetic soft robotics," in *Proc. 28th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS)*, Jan. 2015, pp. 1044–1047.

[18] H. Thien, M. Stommel, F. L. Daheron, A. L. Page, Z. Deng, and W. L. Xu, "Embedded infrared imaging to measure the deformation of a soft robotic actuator," in *Proc. Int. Conf. Image Vis. Comput. New Zealand (IVCNZ)*, Nov. 2016, pp. 1–6.

[19] S. Dirven, M. Stommel, R. Hashem, and W. Xu, "Medically-Inspired Approaches for the Analysis of Soft-Robotic Motion Control," in *Proc. Int. Workshop Adv. Motion Control (AMC)*, Auckland, New Zealand, Apr. 2016, pp. 370–375.

[20] R. Mutlu, G. Alici, and W. Li, "Electroactive polymers as soft robotic actuators: Electromechanical modeling and identification," in *Proc. Int. Conf. Adv. Intell. Mechatron. (AIM)*, Jul. 2013, pp. 1096–1101.

[21] C. Duriez, "Control of elastic soft robots based on real-time finite element method," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2013, pp. 3982–3987.

[22] F. Largillière, E. Coevoet, M. Sanz-Lopez, L. Grisoni, and C. Duriez, "Stiffness rendering on soft tangible devices controlled through inverse fem simulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 5224–5229.

[23] R. Kang, D. T. Branson, E. Guglielmino, and D. G. Caldwell, "Dynamic modeling and control of an octopus inspired multiple continuum arm robot," *Comput. Math. Appl.*, vol. 64, pp. 1004–1016, Sep. 2012.

[24] F. Boyer and A. Belkhiri, "Reduced locomotion dynamics with passive internal DoFs: Application to nonholonomic and soft robotics," *IEEE Trans. Robot.*, vol. 30, no. 3, pp. 578–592, Jun. 2014.

[25] F. Renda, M. Giorelli, M. Calisti, M. Cianchetti, and C. Laschi, "Dynamic model of a multibending soft robot arm driven by cables," *IEEE Trans. Robot.*, vol. 30, no. 5, pp. 1109–1122, Oct. 2014.

[26] F. Renda, V. Cacucciolo, J. Dias, and L. Seneviratne, "Discrete cosserat approach for soft robot dynamics: A new piece-wise constant strain model with torsion and shears," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 5495–5502.

[27] M. Stommel and W. Xu, "Learnability of the moving surface profiles of a soft robotic sorting table," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 4, pp. 1581–1587, Oct. 2016.

[28] M. Stommel and W. Xu, "Optimal, efficient sequential control of a soft-bodied, peristaltic sorting table," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 858–867, Apr. 2016.

[29] Z. Deng, M. Stommel, and W. Xu, "A novel soft machine table for manipulation of delicate objects inspired by caterpillar locomotion," *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 3, pp. 1702–1710, Jun. 2016.

**Martin Stommel**, photograph and biography not available at the time of publication.

**Zhicong Deng**, photograph and biography not available at the time of publication.

**Weiliang L. Xu**, photograph and biography not available at the time of publication.