Escape Behaviour for the Task-Level Control of a Soft Robot

Martin Stommel
Dept. of Electrical & Electronic Engineering
Auckland University of Technology
Auckland, New Zealand
mstommel@aut.ac.nz

Zhicong Deng
Dept. of Mechanical Engineering
The University of Auckland
Auckland, New Zealand
zden012@aucklanduni.ac.nz

Weiliang Xu
Dept. of Mechanical Engineering
The University of Auckland
Auckland, New Zealand
p.xu@auckland.ac.nz

***Abstract*—A soft robotic surface manipulator is a device equipped with a computer-controlled, deformable surface. By moving the surface, objects placed on the robot can be transported, rotated, or otherwise brought into a desired configuration. A probabilistic automaton can use camera feedback to measure and control the movement of the work pieces, and find the optimal sequence of actions to reach a desired state. The probabilistic nature of the method solves most problems resulting from noise and impulsive disturbances. In some cases, however, there are systematical disturbances that are out of the scope of a probabilistic method. In a soft robotic surface manipulator, this can lead to objects becoming pinned against an obstacle or performing repetitive sequences of actions. This paper describes a method to escape from these situations. The method is supported by simulations based on the measured properties of an existing soft table. This is the first publication dedicated to this problem in soft robotic, task-level control.**

***Index Terms*—Soft robotics, mechatronics, planning, control, probabilistic automaton, machine vision**

I. Introduction

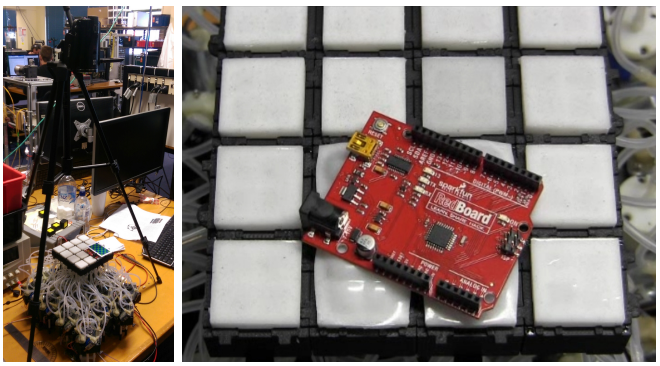
Soft robotics is a new research direction dealing with robots made out of soft materials. Many research questions concerning the mechatronic design, fabrication, and operation are still unanswered. In this paper, we analyse how a soft robot can be controlled using feedback from machine vision. In particular, we are interested in how the vision feedback can be used to navigate the state space of a soft robotic surface manipulator (Fig. 1) for task-level control. It turns out that the sensor limitations and uncertainties of a physical setup can lead to oscillatory behaviour that prevents convergence to a desired state. We present the first algorithm in soft robotics to tackle this problem.

II. Related Works

From the perspective of task-level control, the existing approaches can be classified into (1) the detailed modelling of one or more actuators, (2) the generation of complex but repetitive movements, and (3) the control of the robot behaviour to solve a task.

Geometrical and dynamical models have been developed for elongated structures (robotic tentacles, arms, beams) and 2D surfaces. The aim is to estimate the shape for a set of control signals or, inversely, to find the control signals to create a certain robot shape. The predominant approach is to discretise the structure into small segments, which in turn can be modelled by approaches known from rigid robotics. A deformable 2D surface can be modelled by concatenating sphere segments [1] or sampling the height over ground along a grid [2]. For elongated structures, a backbone curve can be modelled [3]. The discretisation of elongated structures allows to formulate a kinematic chain to predict the robot shape [4], [5]. Mechanical models have been introduced to describe robot deformations that cannot be represented by a piece-wise constant curvature model [6]. It is possible to optimise the shape of a robot without a model just from sensor measurements of its shape. Because of the high number of measurements it is not suitable for real-time [2].

A big advantage of soft robots is, however, that for many applications it is not required to explicitly compute a detailed robot shape. Instead, the soft material allows for a natural adaptation of the robot to the shape of the surroundings. This can be used to create very simple control algorithms to gener-



(a) Camera setup (b) Surface manipulator in action

Fig. 1. Soft robotic surface manipulator: The robot consists of a 4×4 array of pneumatic actuators. Each actuator possesses a soft surface (white) that can shift an object in four directions. A top-mounted camera tracks the work piece (a red circuit board). The image on the left shows checkerboard markers that were used in initial experiments. By now, the tracking works feature based without additional markers. The actuators work by producing a 'bump' on the surface that lifts the object. By exploiting the interaction between four air chambers embedded in each actuator, the top of the bump can be shifted in parallel to the ground. For this study, we measured the stochastical properties of the robot and modelled it by a probabilistic automaton. The automaton is used for path planning in our simulations.

ate robot movements that are useful at application level. A soft gripper, for example, can lift a delicate object without detailed knowledge of its geometry [7]. Crawling [8], [9], walking [10], [11], and swallowing movements [12] have been demonstrated using fixed sequences of control signals in an open-loop manner. The basic movement types of soft robots can be categorised [13] into elongation/shortening, bending, flowing, transformation/reconfiguration. These basic movements can be combined to create higher abilities. Partially based on a review by Laschi [14], we can abstract the following application-level abilities: locomotion (walking, crawling, swimming, jumping, climbing), grasping, arm movement, shape transforms (growing/shrinking, elongation/shortening, reconfiguration), and object manipulation (repositioning/reorientation [1], swallowing, digestion [15], [16]).

For task-level control, the behaviour of the robot must be aimed at the achievement of a task. This requires the sensing of the state of the robot and its environment [17]. By discretising the state space, it is possible to formulate the robot behaviour as a sequence of probabilistic state transitions triggered by realising selected soft robotic abilities. This allows to use a probabilistic automaton [18], [19] and reinforcement learning [20] to calculate the robot behaviour in a data-driven manner. The data-driven approach allows for the effective use of the existing soft robotic abilities without requiring a complex mechanical (and computational) model. Probabilistic state transition account for noise and variations in the behaviour. However, there can be systematic errors that are out of the scope of a probabilistic method and that have not been considered in soft robotics, yet. They are the topic of the present article.

III. Problem Statement

Let $S \subset \mathbb{N}$ be the set of states of the robot (including relevant environmental properties, e.g. the state of a work piece), ${}^n s \in S$ be the state at discrete time n, A be the set of actions that the robot can perform (according to its abilities), and $a \in A$ be a single action. We then characterise the behaviour of the robot by the probabilistic state transitions $\delta : S \times A \times S \mapsto [0\,1]$, where

$$\delta(s_k, a, s_j) = p(s_j | s_k, a). \tag{1}$$

The relation δ describes the development of the system state over time by giving the probability that state ${}^n s = s_k$ at time n changes into a successor state ${}^{n+1} s = s_j$. Let F be a set of final states that model the successful completion of a certain task.

The probability of reaching any $f \in F$ by performing a single action a is given directly by δ. For a sequence of actions, one action can be selected at a time, whereas future actions need to be selected based on how the system state really develops. The probability of reaching any f in maximally L steps is given by

$$\tilde{p}_L(a, s) = \sum_{\tilde{s} \in S} p(\tilde{s} | s, a) \max_{\tilde{a} \in A} \tilde{p}_{L-1}(\tilde{a}, \tilde{s}),\ 1 < L. \tag{2}$$

To select a, we consider the expected number of actions

$$c(s, a) = \sum_{l=1}^{L} l \left(\tilde{p}_l(a, s) - \tilde{p}_{l-1}(a, s) \right),\ 1 < l. \tag{3}$$

that the robot will need to perform until it reaches a final state. Given δ, the best action

$$a_{opt} = \arg\min_{a \in A} c(s, a) \tag{4}$$

can be computed offline for every state of the robot and stored in a look-up-table. Figure 2a shows an example.

During the control of the robot, we measure the current state ${}^n s$. We then perform the action recommended by our look-up-table. The process repeats until the robot is in a desired final state (cf. Fig. 2b).

In the case of a stochastic, transient disturbance, the real successor state can be different from the expected successor state. This is not a problem, because the look-up-table will indicate the best action for any new state. Random delays have already been considered in the computation of the best actions.

However, in a real application there can be systematic, persisting errors that are out of the scope of the probabilistic planning algorithm. Let ${}^0 s, {}^1 s, \ldots, {}^n s$ be the sequence of states visited up to the current state ${}^n s = s_k$ time n. Suppose, a systematic error brings the robot into the successor state ${}^{n+1} s = {}^m s$, where $0 \leq m \leq n$. Since the table of best actions remain unchanged, it is a likely scenario that the robot applies the same sequence of actions (with minor probabilistic variations) until it reaches s_k again. If the systematic error persists, then the robot is set back to ${}^m s$. This results in oscillatory movements preventing the robot from completing its task.

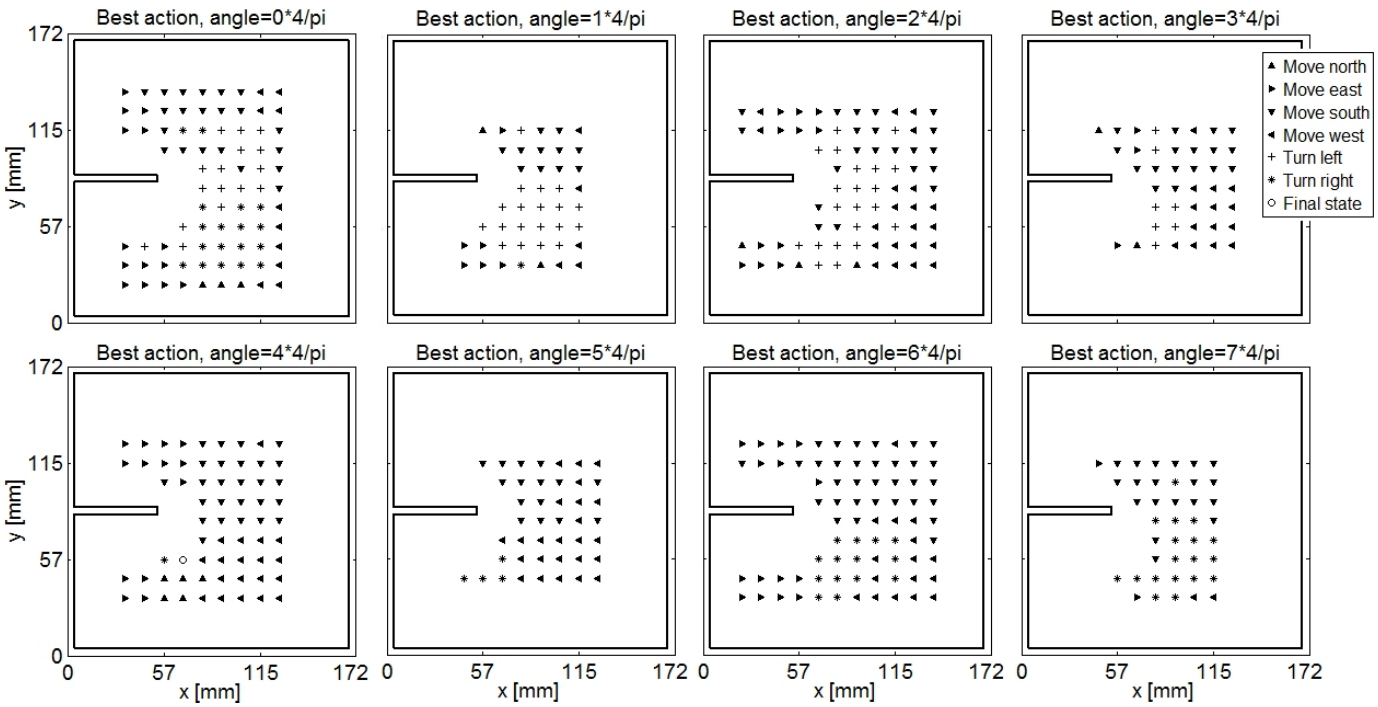
This can for example happen if the soft robot handles an object, but the object becomes entangled in an obstacle. If the exact nature of the entanglement is either not detected or not modelled in the state space, then the robot might keep up the entanglement. Figures 2c and 2d show examples.

This paper proposes a method to recognise this situation and navigate the state space in a way to avoid the systematic error.

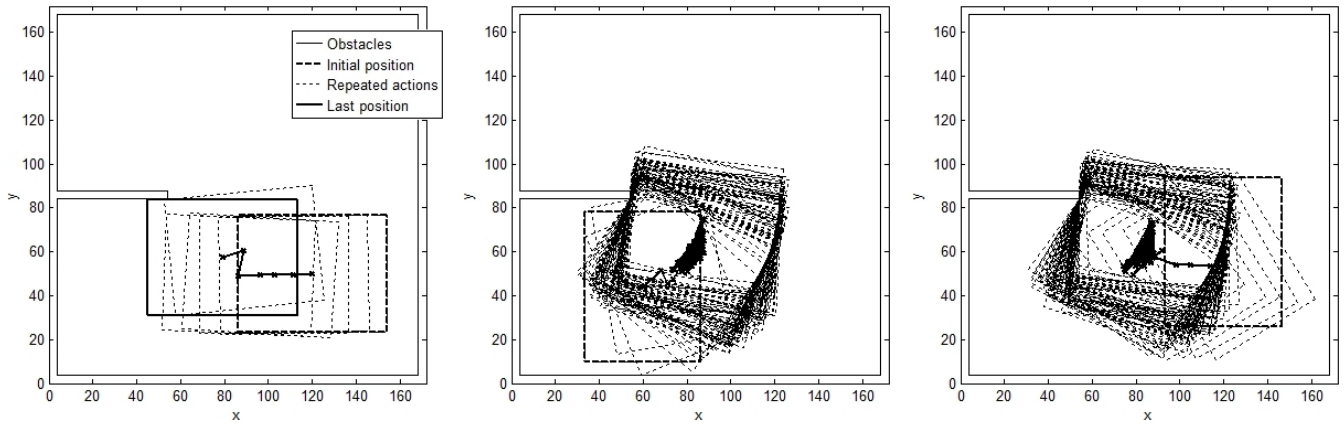
IV. Proposed Method

First, we note that entangled objects are difficult to recognise by sensor readings alone. Not every collision causes entanglement. It is therefore not straightforward to avoid the problem by designing a more informative state model. Oscillations can also be caused if two different situations (e.g. free object + collision) are inaccurately assigned to a joint state. There are, however, no theoretical guidelines to quantise the state space in a way that avoids the problem. A finer quantisation reduces the frequency of the seeing oscillations, but the cause of the problem remains.

Instead of trying to solve the problem from within the framework, we move to another level of abstraction and monitor the history of state transitions while the robot is



(a) Best action



(b) Good trajectory (c) Entanglement (d) Entanglement

Fig. 2. Demonstration of oscillatory behaviour in the control of a soft surface manipulator: The goal of this simulation is to bring the rectangular work piece into a horizontal position in the lower left. The simulation is modelled after the robotic setup shown in Fig. 1. The details of the simulation are described in Sec. V. An additional obstacle on the middle left is modelled to make the scenario more challenging. a) The state space consists of the position (x and y) and orientation θ of the work piece. It is discretised along a regular grid. For every state, the optimal action is computed. b) Usually, the work piece can be moved into the desired state without problems. Stochastical deviations are tolerated and considered in the planning of the best actions. Here, the robot performs 6 actions before reaching the final state. Only one state is reached twice, the other states only once. c,d) The interaction with the obstacle resets the work piece to an earlier state, thereby causing repetitive movements. Here, the simulation is stopped after performing 80 actions without reaching the final state. In example (c), the most frequently visited states were reached 34 and 17 times respectively. In example (d), the most frequently visited states were reached 31 and 12 times respectively.

operating. The basic idea is to maintain a histogram of visited states. If a state is visited too many times, then the robot performs a random walk until it escapes the systematical disturbance. The exact procedure is as follows:

1) Let $h[s]$ denote how many times state s has been visited. Initially, $h[s] = 0$ for all $s \in S$. Let $e \in \{true, false\}$ indicate if there are oscillations. Initially, $e := false$. Let t_{low} and t_{high} be two thresholds with $t_{low} < t_{high}$.
2) At current time n, the state ${}^n s$ is measured.
3) if ${}^n s \in F$, then the algorithm stops (sucessfully).
4) if not e, then the histogram is updated as $h[{}^n s] := h[{}^n s] + 1$ and we check if oscillatory movement has started by setting $e := true$ if $h[{}^n s] > t_{high}$.
5) if e, then we check if the robot escaped the disturbance, i.e. if $h[{}^n s] < t_{low}$, then we set $e := false$ and reset the histogram to $h[s] = 0$ for all $s \in S$.
6) Read $a := a_{opt}$ from the lookup table.
7) if e then a is selected as a random action from $A \backslash \{a_{opt}\}$.
8) The robot performs action a. This increments the time $n := n + 1$ and we continue at step 2.

Because of noise, not all states of an oscillatory movement are visited the same number of times. We model this by using the two thresholds t_{low} and t_{high}. The thresholds depend on the application.

Performing a random action might seem simple. Also, not selecting a_{opt} could indicate that maybe performing the single worst action (instead of the best) might be a good strategy. However, disturbances are likely to occur in confined areas of the state space, e.g. where the movement of the robot or the work piece is restricted. By adopting a deterministic strategy like 'always perform the worst action', there is a chance of bringing the robot repeatedly into another difficult area of the state space. The random strategy leaves more options.

V. Setup of the Experiments and Simulations

We validated the proposed method by the scenario shown in Fig. 1. An existing soft surface manipulator [1] is used to translate (4 directions) and rotate (2 directions) the circuit board shown in Fig. 1b. The position x, y and angle θ of the board is recorded by a top mounted camera. The intrinsic parameters were calibrated using a checkerboard pattern, so the position of the board can be obtained in world coordinates.

To create the set S of states, we discretised a workspace area of 172mm $\times$ 172mm into 15 steps in x and y direction. The workspace area is compatible with an array of 4×4 or 5×5 physical actuators. The orientation was discretised into 8 steps. A rigid border and an additional obstacle reaching into the workspace area from the left side was modelled in software. This creates confined spaces on the surface, where oscillatory behaviour can be demonstrated. An analysis of the theoretically possible 1800 states showed that only 472 states were physically reachable (no intersection of solid bodies).

A set of six actions A was modelled after the real behaviour of the robot. The actions were to move north/east/south/west over 7 cycles of pressurisation [1] and turn left/right over 15 cycles. To define the actions, we recorded 111 video sequences of the moving, unobstructed work piece. The positions before and after performing an action were measured, resulting in covariance matrices of x, y and θ for each action. The covariance matrices were based on 35 to 72 observations per action. On average, 1 action moves the workpiece by about 1cm and 8 actions perform a full rotation.

To model the state transitions, we sampled initial states along a fine grid through the reachable area of the state space. The successor state was then sampled from the measured covariance matrices representing the robot behaviour. These, however, are only valid in unobstructed areas of the robot. Therefore, we simulated a path between the initial and successor position to check for collisions between the work piece and obstacles. In the case of collisions, an alternative trajectory was computed using a simple physical model. The model assumes rotation and deflection around a point of contact. To simplify the simulation, mechanical forces were not considered here. We evaluated about $2 \cdot 10^6$ state transitions, which could be accomplished in 2:48h CPU time on and Intel quad-core CPU with 3.2 GHz. Only a very small number of states was not sufficiently covered by the sampling and had to be removed from the simulation.

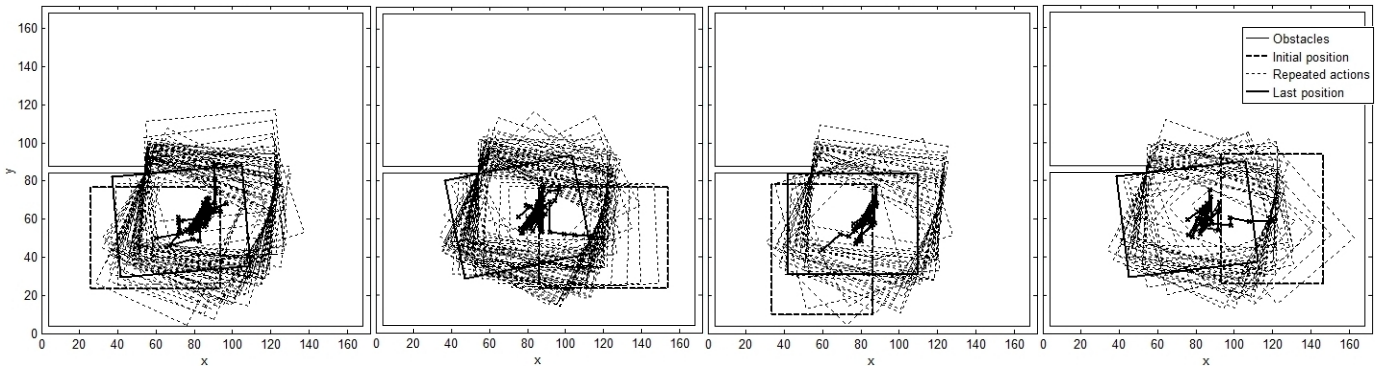
Trajectories of entangled objects were not considered in the computation of the state transition probabilities. Since oscillatory movements correspond to an infinite path length, they would not contribute meaningfully to the probabilistic approach. Instead, a fixed penalty for a manual intervention could be assumed. But since such a penalty would be application dependent and examples of entangled objects are difficult to generate (and application dependent as well), this has not been done here.

We were then able to define initial and final states and plan the best actions for every current state. The planning for path lengths of maximally 25 steps took about 2s CPU time. Since both the sampling of the state transition probabilities and the planning of the best actions are offline steps, the CPU time is uncritical. Because of the design of the table and control task, the best actions and expected successor states form a beautiful minimum spanning tree (cf. Fig. 2a), and the probability of reaching a final state was almost 100 percent for every valid state. This, however, is not generally given, as the trivial (but practically not very relevant) example of an intentionally impossible task shows.

By using the designed states, actions, state transition probabilities, and planning results, we were able to simulate the behaviour of the robot and run the algorithm given in Sec. IV. The thresholds $t_{low} = 3$ and $t_{high} = 4$ were chosen experimentally.

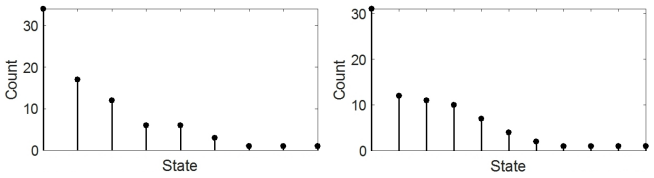
VI. Results

Examples of oscillatory movements are shown in Fig. 2. The workpiece is moved against the obstacle and deflected. This changes the state back to a previously visited state. The tabulated best actions indicate moving the work piece in the direction of the problem. As Fig. 4 shows, most oscillations



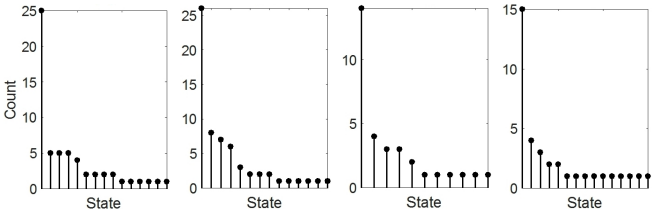
(a) Approach from below (b) Approach from the right (c) Translation and rotation (d) Translation and rotation

Fig. 3. The proposed method disentangles the work piece from the obstacle. For comparability we chose the same final state and best actions as shown in Fig. 2. Also, the initial state of the examples (c) and (d) is the same as in Figs. 2c and 2d. In all examples, the oscillatory movements have been detected and stopped successfully. The final state is reached after performing a finite number of actions.



(a) (b)

Fig. 4. Histogram of visited states for the entangled workpiece shown in Fig. 2. a) The trajectory shown in Fig. 2c covers 9 different states. Since the situation corresponds to an infinite loop, only the relative frequencies are important. Most loops extend over 1–3 states, only. b) The trajectory shown in Fig. 2c covers 11 different states.



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

Fig. 5. Histogram of visited states for the proposed method. The diagrams correspond to the trajectories seen in Fig. 3a–d.

cover varying, short sequences of states with one state being part of most iterations.

Figure 3 shows the result of using the proposed method. In all tested cases, the oscillations could be broken. But as Fig. 5 shows, some states are still heavily visited. One reason is that the proposed method requires a certain state to be visited a number of times (given by threshold t_{high}) before it reacts. The threshold cannot be set too low, because otherwise naturally re-visited states would trigger a random walk. In the examples, the method resolves all detected oscillations within 1–3 state transitions. Since the disturbance persists and the work piece is still close to it, there is a chance that the random walk does not reach the other side of the disturbance. In that case, multiple attempts to escape the disturbance are required. In the examples shown in the Figs. 3a–d, the numbers of attempts were 4, 5, 2, 2, respectively.

VII. Summary

The task level control of soft robots has not been researched much, yet. In the recent years, many new soft actuators have been developed and used to realise powerful soft robotic abilities. But to accomplish task-level control, it is also necessary to monitor the state of the robot and environment in order to plan and recognise the successful completion of a task. This article contributes valuable experience to this area.

For the example of soft robotic surface manipulation, we developed a closed-loop control system at task-level. While the control itself was conducted in simulation, the system is modelled after the properties of a physical prototype.

Our results show that the navigation of the state space is subject to surprising effects that are beyond the control of single actuators or the open-loop control of repetitive locomotion behaviours. While it is trivial that physically unreachable states must be avoided in a software model, they can also be related to disturbances at task-level. The simple transportation of a work piece in a static environment can show oscillations if there are noise and uncertainties.

In this article, we presented an algorithm that frees an entangled workpiece and breaks oscillatory movements. The way out of an entanglement might not always be simple. But the algorithm tries multiple times in a randomised manner until a desired final state is reached.

References

[1] Z. Deng, M. Stommel, and W. L. Xu, "A Novel Soft Machine Table for Manipulation of Delicate Objects Inspired by Caterpillar Locomotion," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 3, pp. 1702–1710, 2016.

[2] M. Stommel and W. L. Xu, "Learnability of the Moving Surface Profiles of a Soft Robotic Sorting Table," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 4, pp. 1581–1587, 2016.

[3] R. Mutlu, G. Alici, and W. H. Li, "Electroactive Polymers as Soft Robotic Actuators: Electromechanical Modeling and Identification," in *International Conference on Advanced Intelligent Mechatronics (AIM)*, July 2013, pp. 1096–1101.

[4] R. Kang, D. T. Branson, and E. G. D. G. Caldwell, "Dynamic Modeling and Control of an Octopus Inspired Multiple Continuum Arm Robot," *Computers & Mathematics with Applications*, vol. 64, pp. 1004–1016, 2012.

[5] F. Boyer and A. Belkhiri, "Reduced locomotion dynamics with passive internal dofs: Application to nonholonomic and soft robotics," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 578–592, June 2014.

[6] 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 *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 5495–5502.

[7] F. Ilievski, A. D. Mazzeo, R. F. Shepherd, X. Chen, and G. M. Whitesides, "Soft Robotics for Chemists," *Angewandte Chemie International Edition*, vol. 50, no. 8, pp. 1890–1895, 2011.

[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 Transactions on Mechatronics*, 2012.

[9] H.-T. Lin, G. G. Leisk, and B. Trimmer, "GoQBot: A Caterpillar-inspired Soft-bodied Rolling Robot," *Bioinspiration & Biomimetics*, vol. 6, p. 026007, 2011.

[10] 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 (PNAS)*, 2011.

[11] D. Morales, E. Palleau, M. D. Dickey, and O. D. Velev, "Electro-actuated hydrogel walkers with dual responsive legs," *Soft Matter*, vol. 10, pp. 1337–1348, 2014.

[12] S. Dirven, W. L. Xu, J. Allen, L. Cheng, and J. Bronlund, "Biologically-Inspired Swallowing Robot for Investigation of Texture Modified Foods," *International Journal of Biomechatronics and Biomedical Robotics*, vol. 2, no. 2–4, pp. 163–171, 2013.

[13] L. Wang and F. Iida, "Deformation in Soft-Matter Robotics: A Categorization and Quantitative Characterization," *IEEE Robotics Automation Magazine*, vol. 22, no. 3, pp. 125–139, Sept 2015.

[14] C. Laschi, B. Mazzolai, and M. Cianchetti, "Soft robotics: Technologies and systems pushing the boundaries of robot abilities," *Science Robotics*, vol. 1, 2016.

[15] Y. Dang, L. K. Cheng, M. Stommel, and W. L. Xu, "Technical Requirements and Conceptualization of a Soft Pneumatic Actuator Inspired by Human Gastric Motility," in *International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*. IEEE, 2016, p. 6.

[16] R. Hashem, W. L. Xu, M. Stommel, and L. K. Cheng, "Conceptualisation and Specification of a Biologically-Inspired, Soft-Bodied Gastric Robot," in *International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*. IEEE, 2016, p. 6.

[17] M. Stommel, W. L. Xu, P. P. K. Lim, and B. Kadmiry, "Robotic Sorting of Ovine Offal - Discussion of a Soft Peristaltic Approach," *Journal of Soft Robotics*, vol. 1, no. 4, pp. 246–254, 2014.

[18] M. Stommel and W. L. Xu, "Optimal, Efficient Sequential Control of a Soft-Bodied, Peristaltic Sorting Table," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 858–867, 2016.

[19] M. Stommel, Z. Deng, and W. L. Xu, "Probabilistic Automata Model of a Soft Robot for the Planning of Manipulation Tasks," *IEEE Transactions on Automation Science and Engineering*, 2017.

[20] V. Vikas, P. Grover, and B. Trimmer, "Model-free control framework for multi-limb soft robots," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 1111–1116.