ADVANCED INTELLIGENT SYSTEMS
Open Access
www.advintellsyst.com

Hierarchical Control and Learning of a Foraging CyberOctopus

*Chia-Hsien Shih, Noel Naughton, Udit Halder, Heng-Sheng Chang, Seung Hyun Kim, Rhanor Gillette, Prashant G. Mehta, and Mattia Gazzola**

Inspired by the unique neurophysiology of the octopus, a hierarchical framework is proposed that simplifies the coordination of multiple soft arms by decomposing control into high-level decision-making, low-level motor activation, and local reflexive behaviors via sensory feedback. When evaluated in the illustrative problem of a model octopus foraging for food, this hierarchical decomposition results in significant improvements relative to end-to-end methods. Performance is achieved through a mixed-modes approach, whereby qualitatively different tasks are addressed via complementary control schemes. Herein, model-free reinforcement learning is employed for high-level decision-making, while model-based energy shaping takes care of arm-level motor execution. To render the pairing computationally tenable, a novel neural network energy shaping (NN-ES) controller is developed, achieving accurate motions with time-to-solutions 200 times faster than previous attempts. The hierarchical framework is then successfully deployed in increasingly challenging foraging scenarios, including an arena littered with obstacles in 3D space, demonstrating the viability of the approach.

1. Introduction

Soft materials have long been seen as key drivers for improving robots' abilities to interact with and adapt to their environments.[1–3] Nonetheless, it is precisely the advantages afforded by physical compliance that, in turn, render soft robots difficult to control. Indeed, material nonlinearities, instabilities, continuum mechanics, distributed actuation, and conformability to the environment all render the control problem challenging.[4,5] In search of potential solutions, roboticists have turned to nature for inspiration,[6] from the investigation of elephant trunks,[7,8] mammalian tongues,[9,10] and inchworms[11–13] to plant tendrils,[14,15] starfish,[16,17] snakes,[18,19] and octopuses.[20–26] The latter, in particular, have received a great deal of attention owing to their unmatched ability to orchestrate multiple soft arms for swimming, crawling, and hunting, as well as manipulating complex objects, from coconut shells to lidded jars.[27,28]

Within this context, a potential solution framework based on hierarchical decomposition is suggested by the unique neurophysiology of the octopus. In contrast to the mostly centralized brain structure of vertebrates,[29] the octopus exhibits a highly distributed neural system wherein two-thirds of its brain lies within its arms.[30] This peripheral nervous system (PNS) is organized into brachial ganglia, colocated with the suckers, and is responsible for low-level sensorimotor tasks and whole-arm motion coordination.[31] Indeed, surgically severed arms are known for being able to execute motor programs such as reaching or recoiling.[32,33] The central nervous system (CNS), composed of the remaining third of the neural tissue, is instead located in the mantle and is thought to be responsible for learning and decision-making by integrating signals from the entire body.[34] This neural architecture is naturally suggestive of a control hierarchy wherein high-level decisions are made in the CNS, executed by the PNS, and finally modulated by the local environment via arm compliance.

Reflecting these considerations, we present a trilevel hierarchical approach to coordination and learning in an octopus computational analog, henceforth referred to as the CyberOctopus. Our framework, as shown in **Figure 1**, decomposes control into central-level, arm-level, and local environment-level. At the central level, executive and coordination decisions (behavioral primitives), such as reaching for food or

C.-H. Shih, H.-S. Chang, S. H. Kim, P. G. Mehta, M. Gazzola
Department of Mechanical Science and Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA
E-mail: mgazzola@illinois.edu

N. Naughton
Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA

U. Halder
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA

R. Gillette
Department of Molecular and Integrative Physiology
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aisy.202300088.

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

DOI: 10.1002/aisy.202300088

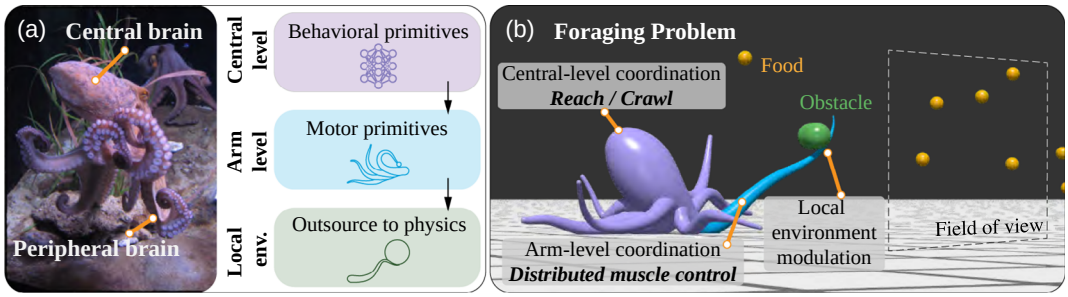

Figure 1. a) Our proposed control hierarchy inspired by the distributed neurophysiology of the octopus: a centralized decision-maker selects appropriate motion primitives, arm-level controllers generate the necessary muscle activations and physical compliance accommodates environmental obstacles. Anneli Salo [CC BY-SA 3.0]. b) A CyberOctopus foraging for food in the presence of obstacles.

crawling, are taken and issued to the individual arms. This top level is implemented as a compact, feedforward neural network. At the level of the arm, modeled as an elastic slender filament, muscle activations (motor primitives) realize incoming commands and produce appropriate deformations.[25] Muscle control is obtained via a fast energy-shaping technique that minimizes energy expenditure.[35] We supplement this control with distributed, local behavioral rules that conspire with the arm's compliant physics to autonomously accommodate for solid environmental features.

The CyberOctopus is shown to learn to forage for food in increasingly challenging scenarios, including an arena littered with obstacles in 3D space. Overall, this work illustrates how hierarchical control is not only viable in soft multiarms systems but, in fact, can significantly outperform end-to-end, deep learning approaches.

2. The CyberOctopus Model

Of all the elements that comprise a real octopus, here, we focus on its arms and their coordination. The arms of the CyberOctopus (**Figure 2**) are modeled as linearly tapered Cosserat rods, which are slender, one-dimensional elastic structures that can undergo all modes of deformation—stretch, bend, twist, and shear—at every cross-section.[36] Each arm is then represented as an individual passive rod upon which virtual muscles produce forces and couples. We consider each arm deforming in-plane on account of two longitudinal muscle groups (LM1 and LM2) and one set of transverse muscles (TM), reflecting the octopus' physiology, as shown in Figure 2a,b. Longitudinal muscles (Figure 2c) are located off-center from the arm's axis of symmetry but run parallel to it, generating both forces and couples that can cause the arm to contract (symmetric co-contractions) or bend (asymmetric co-contractions). Transverse muscles (Figure 2d) are located along the arm's axis but are oriented orthogonally, so that their contraction causes the arm to extend due to incompressibility. Since only planar deformations are considered, we do not model the oblique muscles, which wind around the longitudinal and transverse muscles and whose main function is to provide twist.[37] Twist alone is in fact not relevant to 2D motions, while it becomes important in 3D settings.[38]

2.1. Kinematics

In the Cosserat rod formalism (Figure 2b), each arm is described by its midline position vector $x(s,t) \in \mathbb{R}^2$ within the plane

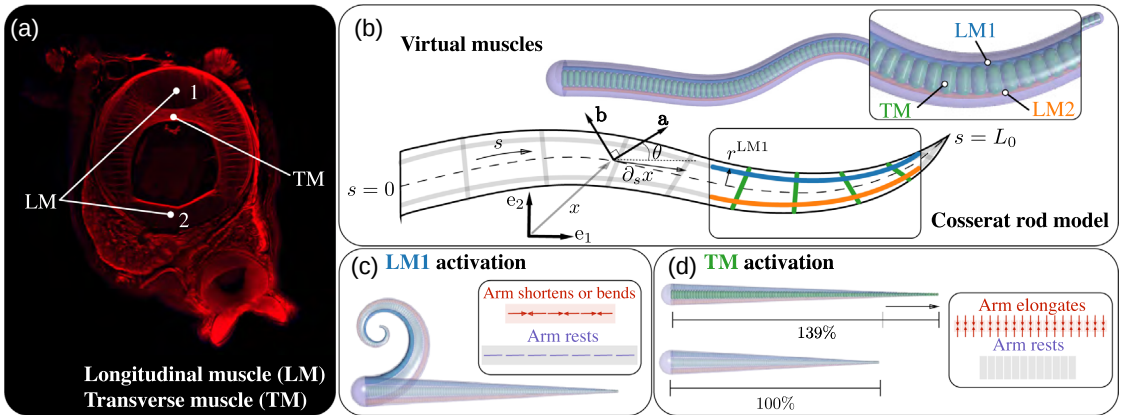

Figure 2. a) Histological cross-section of an *Octopus rubescens* arm showing the longitudinal (LM1 and LM2) and transverse (TM) muscles. Muscles are labeled in red by phalloidin staining (Image credit: Tigran Norekian and Ekaterina D. Gribkova). b) Accordingly, our model arm consists of top (LM1, blue) and bottom (LM2, orange) virtual longitudinal muscles as well as virtual transverse muscles (TM, green). The soft arm itself is represented as a single Cosserat rod, to capture its passive elastic mechanics. Muscle activations are defined along the arm. c) Longitudinal muscle activations result in arm shortening (symmetric co-contraction) or bending (asymmetric co-contraction), while d) transverse muscle activations result in arm elongations, due to tissue incompressibility.

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

spanned by the fixed orthonormal basis $\{\mathrm{e}_1, \mathrm{e}_2\}$ and along the arclength $s \in [0, L_0]$, where L_0 is the arm's rest length, and t is time. The arm's local orientation is described by the angle $\theta(s,t) \in \mathbb{R}$, which defines the local orthonormal basis $\{\mathbf{a}, \mathbf{b}\}$ with $\mathbf{a} = \cos\theta \mathrm{e}_1 + \sin\theta \mathrm{e}_2$ and $\mathbf{b} = -\sin\theta \mathrm{e}_1 + \cos\theta \mathrm{e}_2$. The local deformations of the arm—stretch (ν_1), shear (ν_2), and bending (κ)—are defined by the kinematics of the arm

$$\partial_s x = \nu_1 \mathbf{a} + \nu_2 \mathbf{b}, \quad \partial_s \theta = \kappa \tag{1}$$

For a straight arm at rest, $\nu_1 = 1$ and $\nu_2 = \kappa = 0$.

2.2. Dynamics

The dynamics of the planar arm[25,39] read

$$\partial_t \begin{bmatrix} \rho A \nu_1 \\ \rho A \nu_2 \\ \rho I \omega \end{bmatrix} = \begin{bmatrix} \partial_s \left(\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} \right) \\ \partial_s m + \nu_1 n_2 - \nu_2 n_1 \end{bmatrix} - \zeta \begin{bmatrix} \nu_1 \\ \nu_2 \\ \omega \end{bmatrix} \tag{2}$$

where $\nu = (\nu_1, \nu_2)$ and ω are the linear and angular velocity of the arm, respectively, ρ is the density of the arm, A and I are the arm's local cross-sectional area and second moment of area, $n = (n_1, n_2)$ and m are the internal forces (in the local frame) and couples along the arm, and $\zeta > 0$ is a damping coefficient capturing viscoelastic effects. The arm dynamics (Equation (2)) are accompanied by a set of fixed/free-type boundary conditions for all $t \geq 0$

$$\begin{aligned} &\text{(fixed)} && x(0,t) = x_0, \quad \theta(0,t) = \theta_0 \\ &\text{(free)} && n(L_0,t) = 0, \quad m(L_0,t) = 0 \end{aligned} \tag{3}$$

where $x_0 \in \mathbb{R}^2$ and $\theta_0 \in \mathbb{R}$ are the prescribed position and orientation of the arm at its base. Since the arm is freely moving, a free boundary condition at the tip is chosen.

2.3. Internal Stresses

The overall internal forces and couples acting on the arm (n, m) encompass both passive and active effects

$$n = n^{\mathrm{e}} + \sum_{\mathrm{m} \in M} n^{\mathrm{m}}, \quad m = m^{\mathrm{e}} + \sum_{\mathrm{m} \in M} m^{\mathrm{m}} \tag{4}$$

where $(n^{\mathrm{e}}, m^{\mathrm{e}})$ are restoring loads due to passive elasticity, and $(n^{\mathrm{m}}, m^{\mathrm{m}})$ are the active loads resulting from the contraction of muscle $\mathrm{m} \in M$, with $M = \{\mathrm{LM1}, \mathrm{LM2}, \mathrm{TM}\}$ being the collection of muscle groups in the arm.

For a linearly elastic arm, passive elastic forces and couples read

$$n^{\mathrm{e}} = \begin{bmatrix} EA(\nu_1 - 1) \\ GA\nu_2 \end{bmatrix}, \quad m^{\mathrm{e}} = EI\kappa \tag{5}$$

where E and G are Young's and shear moduli.

The contraction of a muscle $\mathrm{m} \in \{\mathrm{LM1}, \mathrm{LM2}, \mathrm{TM}\}$ is modeled via the activation function $\alpha^{\mathrm{m}}(s,t) \in [0,1]$, with 1 corresponding to maximum activation (Figure 2c,d). When virtual muscles contract, they produce on the arm a distributed force $n^{\mathrm{m}}(s,t)$ (in the local frame) that depends on the musculature's spatial organization. We model this as

$$n^{\mathrm{m}} = \begin{bmatrix} \alpha^{\mathrm{m}} \sigma^{\mathrm{m}} A^{\mathrm{m}} \mathrm{T}^{\mathrm{m}} \\ 0 \end{bmatrix} \tag{6}$$

where σ^{m} is the maximum stress generated by the muscle, A^{m} is its cross-sectional area, and T^{m} accounts for the musculature configuration. For longitudinal muscles, $\mathrm{T}^{\mathrm{LM}} = 1$ since contractions directly translate into compression forces along the arm (note that due to small shear $\nu_2 \approx 0$—confirmed numerically—the vector $\mathbf{a}$ of Figure 2b effectively coincides with $\partial_s x$). For transverse muscles, $\mathrm{T}^{\mathrm{TM}} = -1$, capturing the fact that their contractions cause radial shortening, which in turn extends the arm due to incompressibility (Figure 2d). Additionally, the muscles' force–length relationships are modeled here as a constant function, however, more complex descriptions can be incorporated.[25] The muscle's cross-sectional areas, and thus the contraction forces generated along the arm, are assumed to be proportional to the arm's cross-sectional area, and so they are larger at the base and decrease linearly as the arm tapers toward the tip.

Due to the offset location of the longitudinal muscles with respect to the arm's main axis, the active forces n^{LM} generate couples (m^{LM}), which are modeled as follows. We denote the position of a muscle relative to the arm centerline by the vector $x^{\mathrm{m}}(s) = \pm\phi^{\mathrm{m}}(s)\mathbf{b}$, where $\phi^{\mathrm{m}}(s)$ is the off-center distance (Figure 2b). The positive and negative signs are associated with LM1 and LM2, respectively. Then the resulting couples are

$$m^{\mathrm{m}} = (x^{\mathrm{m}} \times n^{\mathrm{m}}) \cdot (\mathrm{e}_1 \times \mathrm{e}_2) = \pm\phi^{\mathrm{m}} \alpha^{\mathrm{m}} \sigma^{\mathrm{m}} A^{\mathrm{m}} \mathrm{T}^{\mathrm{m}} \tag{7}$$

Transverse muscles are arranged perpendicularly to the arm (Figure 2d), and thus result in no couples ($m^{\mathrm{TM}} = 0$).

2.4. Static Configurations

For a static muscle activation $\alpha^{\mathrm{m}}(s)$, the equilibrium configuration of the arm is characterized by the balance of forces and couples. This is obtained by equating the left-hand side of the dynamics (Equation (2)) to zero, yielding

$$\underbrace{\begin{bmatrix} n^{\mathrm{e}} \\ m^{\mathrm{e}} \end{bmatrix}}_{\substack{\text{Elastic arm} \\ \text{passive response}}} = -\underbrace{\sum_{\mathrm{m} \in M} \begin{bmatrix} n^{\mathrm{m}} \\ m^{\mathrm{m}} \end{bmatrix}}_{\substack{\text{Muscle} \\ \text{contractions}}} \tag{8}$$

Equation (8) is solved for the static strains ν_1, ν_2, κ, which in turn lead to the equilibrium configuration of the arm, obtained by integrating the kinematics of Equation (1).

While Equation (8) suffices to determine the equilibrium configuration for given muscle activations $\alpha^{\mathrm{m}}(s)$, it does not account for the dynamical response of the arm transitioning between activations, all the while experiencing environmental loads. To remedy this, we evaluate the effect of muscle activations (and thus of the control policies that determine them) in Elastica,[36,40,41] an open-source software for simulating the dynamics of Cosserat rods (Equation (2)). Elastica has been demonstrated across a range of biophysical applications from soft[41] and biohybrid[40,42–44] robots to artificial muscles[45] and

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

biolocomotion.[18,25,35,40] In Elastica, our CyberOctopus consists of a head and eight arms, of which only a subset (gradually increased throughout the paper) is actively engaged. Material and geometric properties of our model octopus are determined from typical literature values[46] as well as experimental characterizations of *Octopus rubescens*.[35] Numerical values and details of our muscle models are provided in Table S1–S3, Supporting Information, and in the study of Chang et al.[25] Contact forces between the arm and environmental obstacles are modeled using a frictionless force-displacement condition,[36] as described in the Supporting Information.

3. Arm-Level Problem: Motor Execution

Octopuses perform certain goal-directed arm motions via templates of muscle activations, such as traveling waves of muscle contractions.[32] These templates are encoded into the arm's peripheral nervous system as low-level motor programs that are selected, modulated, and combined together to achieve basic behaviors such as reaching and fetching.[32,47,48] Inspired by this, we define two types of primitives for inclusion in our hierarchical approach: motor primitives (Section 3.1) and behavioral primitives (Section 4.1). Motor primitives are low-level motor programs that coordinate the contraction of the CyberOctopus' muscles to accomplish a stereotypical motion. Behavioral primitives are sequences of motor primitives whose combination enables the completion of simple goal-directed tasks (here crawling or reaching available food). These behavioral primitives can then be further composed into more complex behaviors, such as foraging.

3.1. Motor Primitives: Reaching to a Point in Space

We focus on a motor primitive that efficiently moves the tip of the arm to a specified location $q \in \mathbb{R}^2$. By concatenating sequences of this motor primitive, a variety of tasks, such as reaching to a food target, fetching food to the mouth, or crawling can be attained. While in the present context this single motor primitive is sufficient to coordinate planar arm deformations, in general more than one primitive may be desirable (e.g., 3D deformations),[38,49] in which case our approach can be readily extended.

3.1.1. Energy Shaping (ES)

To effect this motor primitive, we employ the energy shaping methodology.[50–52] As developed in our prior work,[25,35,49] an energy-shaping control law is derived to determine the static muscle activations $\alpha = \{\alpha^{\mathrm{m}}\}_{\mathrm{m}\in M}$ that cause the tip of the arm to reach a target location. The equilibrium arm configuration that achieves this goal is obtained by solving an optimization problem that minimizes the tip-to-target distance $\delta(\alpha, q) = |q - x(L_0)|$ along with the muscle activation cost[25]

$$\text{Energy Shaping (ES)} \left\{ \underset{\alpha(\cdot);\alpha^{\mathrm{m}}(s)\in[0,1]}{\text{minimize}} \quad J(\alpha;\alpha_0,q) = \overbrace{\int_0^{L_0} |\alpha(s)-\alpha_0(s)|^2 \mathrm{d}s}^{\text{muscle cost}} + \overbrace{\mu_{\text{tip}}\delta(\alpha,q)}^{\text{task specific cost}} \right. \tag{9}$$

where α_0 are the initial muscle activations, and μ_{tip} is a constant (regularization) coefficient. The tip-to-target distance $\delta(\alpha, q)$ is computed using the kinematic constraints of Equation (1) and equilibrium constraints of Equation (8).

3.1.2. Fast Neural Network Energy Shaping (NN-ES)

While we previously demonstrated the use of ES for muscle coordination in a soft arm,[25,35,49] the solution to the above optimization problem is reliant on a computationally expensive forward–backward iterative scheme. Here, in a hierarchical context where energy shaping will be frequently called upon by a high-level controller, fast solutions are instead imperative. In response to this need, we replace the forward–backward scheme with a neural network and directly learn the mapping $\pi: \{q, \alpha_0(s)\} \mapsto \alpha(s)$ that takes initial muscle activations $\alpha_0(s)$ and target location q, and outputs the activations $\alpha(s)$ that cause the arm to reach q while minimizing muscle costs (**Figure 3**a).

In the CyberOctopus' arm, muscle activations are continuous over $s \in [0, L_0]$, requiring us to first obtain a finite-dimensional representation of the activations for use with our neural network, which we accomplish via a set of K orthonormal basis functions $\{e_k(s)\}_{k=1}^K$. The procedure for finding this set is described in the Supporting Information. In this basis set, the continuous muscle actuation profile $\alpha(s)$ is represented by the coefficients $\{\hat{\alpha}_k\}_{k=1}^K$, so that $\alpha(s) = \sum_k \hat{\alpha}_k e_k(s)$. The inputs to the network are then the coefficients of the initial actuation profile $\{\hat{\alpha}_{0,k}\}$ along with the target location q. Denoting the network weights as v, the outputs of the network are then the coefficients of the desired muscle activation profile $\{\hat{\alpha}_k(\mathsf{v})\}$. The loss function of the network is then obtained by recouching Equation (9) as a function of the network weights v

$$\text{Neural network energy shaping (NN-ES)} \left\{ \underset{\mathsf{v}}{\text{minimize}} \quad J\left(\sum_{k=1}^K \hat{\alpha}_k(\mathsf{v}) e_k(s); \sum_{k=1}^K \hat{\alpha}_{0,k} e_k(s), q\right) \right. \tag{10}$$

Individual muscles' activation bounds ($\alpha^{\mathrm{m}}(s) \in [0,1]$) of Equation (9) are enforced via the function $\max(\min(\alpha^{\mathrm{m}}(s), 0), 1)$.

3.2. An Arm Reaching for Food

To enable our arm, we train the mapping π, represented as a feedforward neural network with three hidden layers of 128 rectified linear unit (ReLU) activation functions. This process can be summarized as follows. The network is trained for 4000 epochs. For each epoch, 100 training samples are generated with initial activations $\alpha_0(s)$ randomly selected from a Gaussian distribution, and target locations q (food) randomly selected from a uniform

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

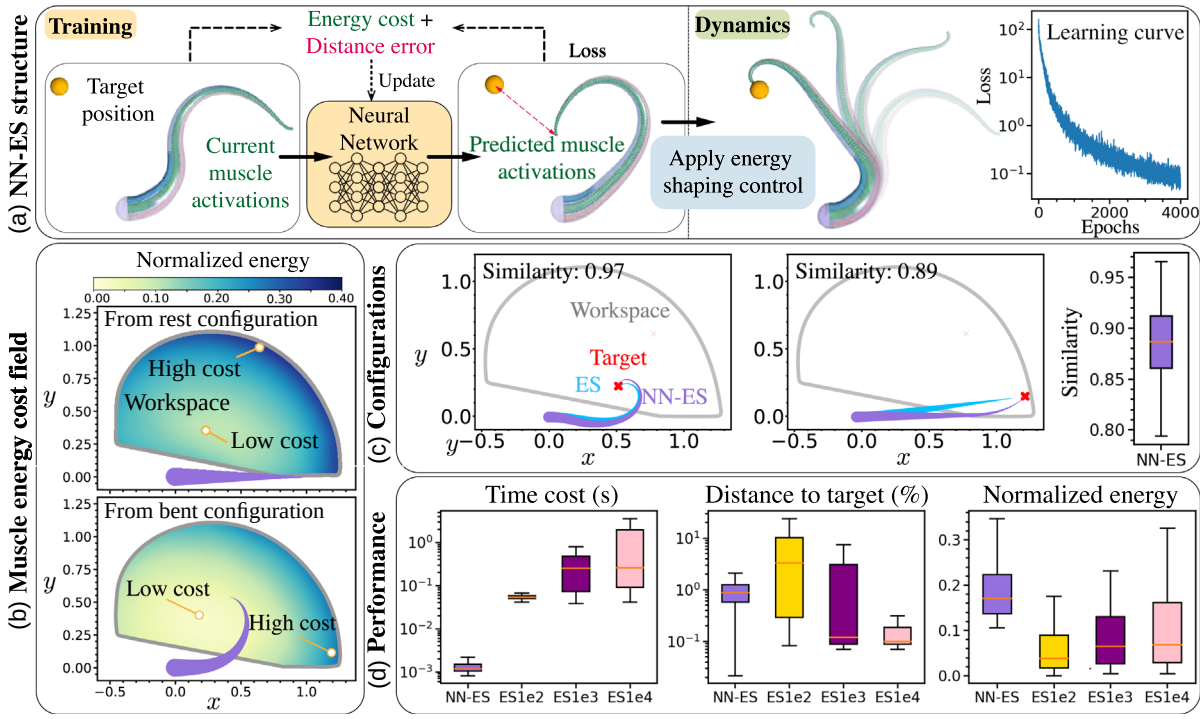

Figure 3. Arm-level controller: a) Neural network energy shaping (NN-ES) control utilizes a learned mapping to determine static muscle activations, and dynamically brings the arm to a given food target. The mapping, represented as a neural network, is trained to take as inputs the food target location and current muscle activations, and then outputs muscle activations that minimize tip-to-food distance and energy expenditure. b) Muscle energy-cost (Equation (9)) normalized by arm length shows the cost of NN-ES to reach a point within the workspace $\mathcal{W}$ given the starting arm configuration (top panel–initially straight arm, bottom panel–initially bent arm). c) NN-ES obtained solutions have high similarity relative to iterative ES solutions.[25] Differences may arise when targets are located toward the edge of the workspace. Indeed, arm configurations obtained via NN-ES are observed to bend near the tip, while ES solutions bend closer to the base. Right panel shows box plot of similarity score between NN-ES and iterative ES solutions, with the orange line representing the median, purple box representing the interquartile (middle 50%) range, and whiskers denoting the min and max of 100 evaluation samples. d) Performance of NN-ES and iterative ES, the latter considering an increasing number of iterations in termination condition. Orange lines represent the metric's median value, boxes represent the interquartile (middle 50%) range, and whiskers denote the min and max of 100 evaluation samples. Comparison shows NN-ES achieves solutions over 200× faster than the iterative ES scheme, while achieving median tip-to-food distances (normalized by the arm length) of less than 1%. The median muscle energy cost of iterative ES increases with the max number of iterations allowed. This reflects the fact that obtained solutions improve with more iterations, decreasing the tip's distance to the target, often through bending and thus higher actuation costs. NN-ES has slightly higher median energy cost (and similar maximum energy cost) as the most accurate iterative ES solution.

distribution over the workspace $\mathcal{W}$ (the set of all points reachable by the tip of the arm). For each training sample, the neural network produces an $\alpha(s, \mathsf{v})$, from which the target-to-tip distance $\delta(\alpha(s, \mathsf{v}), q)$ is computed based on the resulting equilibrium configuration (Equations (8) and (1)). Because δ directly depends on the neural network weights v—through $\alpha(s, \mathsf{v})$—we can compute the gradient of Equation (10) with respect to v and thus update π in an unsupervised manner.

As shown in Figure 3a, the network successfully learns, minimizing the loss function of Equation (10). Exploring the characteristics of the learned mapping, we find that the initial configuration of the arm plays a substantial role in determining muscle activation costs (Figure 3b). For a straight initial configuration ($|\alpha_0(s)|^2 = 0$), targets in the middle of the workspace require less change in muscle activation. Indeed, the arm can reach these targets by activating only the longitudinal muscles along the arm's distal end, which is thinner and hence less stiff. In contrast, targets at the boundary of $\mathcal{W}$ require recruiting both longitudinal and transverse muscles to bend the base (thick and stiff) and extend the arm. For a bent initial configuration, the change in muscle energy is generally lower since longitudinal muscles are already partially activated. This is particularly true for reaching the center of the workspace. The role of the arm's taper in facilitating successful reaching is explored in the Supporting Information.

We next proceed to compare our NN-ES approach with the original iterative ES,[25] employing the termination conditions of normalized tip-to-target distance $\delta(\alpha, q)/L_0 < 0.01\%$ or 10 000 maximum iterations. To quantify differences in the obtained equilibrium configurations $x(s)$, we introduce the similarity metric $D = (1 + \int_0^{L_0} |x^{\text{NN-ES}}(s) - x^{\text{ES}}(s)| \text{d}s)^{-1} \in (0, 1]$, where 1 indicates identical solutions. As seen in Figure 3c, for 100 randomly generated cases, NN-ES and ES produce solutions characterized by a high degree of similarity (average $D = 0.89$). Differences appear for targets located far from the base, with NN-ES stretching and bending the arm toward the tip, while ES directly orienting the entire arm in the food direction. Both algorithms are accurate in reaching food, achieving

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

median tip-to-target distances of less than 1% relative to the rest length of the arm, although we note that NN-ES tends to utilize slightly larger muscle activations than ES (Figure 3d). This drawback is nonetheless compensated by a significant reduction in solution time (Figure 3d), whereby NN-ES outperforms ES by a factor of 200. Further, we note that while ES performance may depend on the allowed maximum number of iterations, the trends described above persist as we span from 100 to 10 000 max iterations. Taken together, these results demonstrate NN-ES to be fast and accurate in coordinating muscle activity and in executing low-level reaching motions. We thus conclude that NN-ES is suitable to be integrated into our hierarchical approach.

4. Central-Level Problem: Coordinating Foraging Behavior

We now turn to the problem of a CyberOctopus foraging for food within a two-dimensional (planar) arena (Figure 1b). Inspired by real octopuses coordinating their arms to move and collect food,[28,53] the CyberOctopus is tasked with maximizing the energy intake derived from collecting food, while minimizing the muscle activations required to reach for it. By engaging its multiple arms, the CyberOctopus can move in any planar direction without reorienting its body. This multidirectionality, compounded by the difficulties associated with distributed muscular actuations across multiple arms, muscle expenditure estimation, limited workspace, and potential presence of solid obstacles, renders the foraging problem challenging.

Here, we define the behavioral primitives available to the central-level controller for orchestrating foraging behavior (Section 4.1), before providing the full problem's mathematical formulation (Section 4.2). Finally, we describe a reinforcement learning-based approach (Section 4.3) that utilizes a spatial attention strategy to simplify the planning process, allowing us to successfully control the CyberOctopus.

4.1. Behavioral Primitives: *Reach All* and *Crawl*

The combination of low-level motor primitives into ordered sequences allows us to construct basic behavioral primitives. We define *reach all* and *crawl*, in an attempt of abstracting out the complexity of foraging into simple terms. Nonetheless, our decomposition approach conveniently provides the opportunity and freedom of defining arbitrary command sets, and different choices may be made.

The *reach all* behavior consists in the arm attempting to reach all food targets within its workspace $\mathcal{W}$. Food is collected sequentially with the ordering determined in a greedy manner. The arm calculates—via the NN-ES controller—the change in muscle activation needed to reach each food target from its current configuration, and then collects the target that requires the least change. This process is repeated until all food in $\mathcal{W}$ is collected (or attempted to). Interaction between the arm and food is not explicitly modeled, with food considered automatically collected once the arm reaches its location.

The *crawl* behavior consists of a predefined set of muscle activations $\alpha^{\mathrm{TM}}_{\mathrm{crawl}}$. First, transverse muscles are activated to extend the arm horizontally by a fixed amount Δr along the crawling substrate. After extension, suckers at the tip of the arm engage with the substrate and transverse muscles are relaxed, pulling the octopus forward by the amount Δr, at which point the suckers are released. We note that even though we do not explicitly model the suckers, their effect is accounted for by appropriately choosing the arm boundary conditions.

We emphasize that these behavioral primitives, while both predicated on the low-level energy shaping motor primitive, are algorithmically distinct, with *reach all* concatenating several motor primitive executions together and *crawl* consisting of a single, predefined execution instead. *Reach all* and *crawl* are then selected and combined by the higher-level controller to generate a variety of foraging behaviors.

4.2. Mathematical Formulation of the Foraging Problem in Hierarchical Form

Let us consider a CyberOctopus with I active arms aiming to collect T food items that are scattered randomly throughout an arena. Food can be found at any vertical location, while the horizontal coordinates are constrained to a two-dimensional Cartesian grid formed by discrete crawling steps (**Figure 4**a). With respect to the bases of the arms, the location of the j-th target is denoted as $q_j \in \mathbb{R}^3$. If the CyberOctopus has complete knowledge of all T food item's locations, the problem for the CyberOctopus is to create an optimal plan that sequentially composes the behavioral primitives—*reach all* and *crawl*—for all of the I arms, so as to fetch all of the food, while minimizing muscle energy expenditure.

Before we model the full problem mathematically, it is useful to examine the simplest case of only one active arm where all targets are within the arm's workspace $\mathcal{W}$. In this case, the CyberOctopus can simply execute a single *reach all* behavior to gather all the targets. However, in cases where some of the targets lay outside $\mathcal{W}$, the CyberOctopus will also need to crawl. Depending on the location of the targets, it may be beneficial for the CyberOctopus to *crawl* even if there are already targets in its workspace. Doing so may serve to bring additional targets within reach, to gather them all more efficiently with a single *reach all* maneuver.

We formulate this optimal planning problem as a Markov decision process (MDP). Even though time has been continuous so far, for the MDP model we consider a discrete-time formulation. Temporal discretization naturally arises by considering the time elapsed between the start and the end of a primitive. Since the motor primitives are nested inside a behavioral primitive (as described in Section 4.1 and Figure 4b), we introduce two different discrete-time indices, at the behavioral (n) and at the motor primitive level ($\hat{n}$).

Thus, at time n, each arm executes a behavioral primitive (action) $u_i[n]$, $i = 1, \ldots, I$, which takes value in the set $\{reach\ all, crawl\}$. This means that each individual arm is treated as functionally equivalent, reflecting the observed bilateral symmetry of octopus arms,[54] and tries to either *reach all* available food targets in its workspace $\mathcal{W}_i$, or *crawls* a fixed amount d_i along its own direction (where $|d_i| = \Delta r$). As a consequence, multiple reaching planes become simultaneously accessible, rendering

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

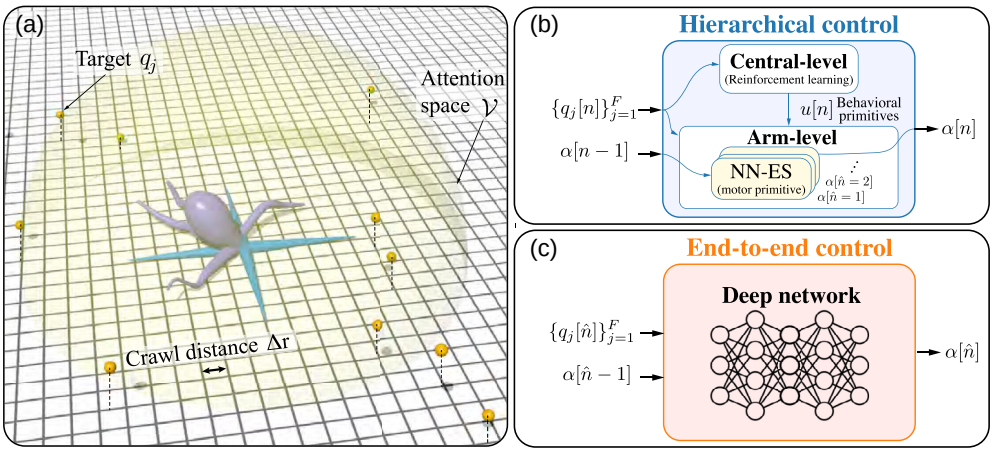

Figure 4. a) Setup of the foraging problem showing the CyberOctopus surrounded by food targets arranged on a grid whose spacing matches the distance traveled during one *crawl* step. The yellow dome shows the attention space heuristic described in Section 4.3, so that the CyberOctopus only considers food items within this space. b) Block diagram for hierarchical controller (Section 4.2, 4.3) and c) end-to-end controller (whose mathematical formulation can be found in the Supporting Information). Both approaches make use of the spatial attention heuristic, and receive the same inputs to produce one activation output. However, the hierarchical decomposition allows us to rewire internally the flows of information, separating concerns into qualitatively different tasks, which can then be efficiently solved by appropriate algorithms.

the foraging problem three-dimensional. Taking the actions of all arms together, the decision variable at time n is denoted as $u[n] = \{u_i[n]\}_{i=1}^{I}$.

The state at time n is denoted as $z[n]$. Its components are

$$z[n] = \{q_j[n], f_j[n]\}_{j=1}^{\mathrm{T}} \tag{11}$$

where the flag $f_j[n] \in \{0, 1\}$ signifies whether the j-th target has already been collected ($f_j[n] = 0$) or not ($f_j[n] = 1$). The dynamics then become

$$\begin{aligned} q_j[n+1] &= q_j[n] - \sum_{i=1}^{I} d_i \mathbb{1}(u_i[n] = \mathit{crawl}) \\ f_j[n+1] &= f_j[n] \prod_{i=1}^{I} (1 - \mathbb{1}(u_i[n] = \mathit{reach\ all}\ \&\ q_j \in \mathcal{W}_i)) \end{aligned} \tag{12}$$

where $\mathbb{1}(\cdot)$ is the indicator function. Here, the first equation expresses the change in the positions (relative to the arm base) of food targets after a *crawl* step, while the second term denotes the change in collection status of food targets after a *reach all* step. When executing the selected behavioral primitives, the CyberOctopus first reaches with any arm that selected *reach all*, before performing any crawling. Finally, the CyberOctopus is not allowed to crawl through the boundaries of the arena. If a boundary is approached, the CyberOctopus remains in place unless it chooses to crawl alongside or backtracks.

Subject to the dynamics of Equation (12), the CyberOctopus aims to find the optimal sequence of behavioral primitives $\overline{u} := \{u[0], u[1], u[2], \ldots, u[N-1]\}$ so as to maximize the cumulative reward

$$\begin{aligned} \max_{\overline{u}} \quad & J(\overline{u}) = \sum_{n=0}^{N-1} R(z[n], u[n]) \\ \text{subject to} \quad & \text{the dynamics (Equation (12)) and given } z[0] \end{aligned} \tag{13}$$

where N is a given stopping time.

The reward function is chosen to capture the trade-off between negative energy expenditure associated with muscle activation, and positive energy associated with food collection. It is defined as

$$\begin{aligned} R(z[n], u[n]) = &-\Phi[n] \\ &+ \sum_{i=1}^{I} \begin{cases} -E^{\mathrm{c}} & \text{if } \mathrm{u_i}[n] = \mathit{crawl} \\ -E_i^{\mathrm{r}}[n] + \gamma f_{\mathcal{W}_i}[n] & \text{if } \mathrm{u_i}[n] = \mathit{reach\ all} \end{cases} \end{aligned} \tag{14}$$

where E^{c} and $E_i^{\mathrm{r}}[n]$ are the total muscle activation costs of the multiple low-level motor primitives necessary to complete the selected command $u_i[n]$ for the i-th arm, γ is the energy of an individual food target, $f_{\mathcal{W}_i}[n]$ is the total number of food items collected during a *reach all* execution by the i-th arm, and $\Phi[n]$ is a penalty term if all I active arms choose *reach all* when no food is available.

If $u_i[n] = \mathit{reach\ all}$ is selected, then the muscle activation sequence $\{\alpha_i[\hat{n}=1], \alpha_i[\hat{n}=2], \ldots, \alpha_i[\hat{n}=\hat{f}_{\mathcal{W}_i}[n]]\}$ is generated for reaching all food targets in the workspace $\mathcal{W}_i$ (as described in Section 4.1, where $\hat{n}$ denotes the discrete-time at the motor level, and $\hat{f}_{\mathcal{W}_i}[n]$ is the total number of food items within reach at time n). If instead $u_i[n] = \mathit{crawl}$, then the predefined activation $\alpha_{\mathrm{crawl}}^{\mathrm{TM}}$ is recruited. Corresponding muscle activation costs are defined as

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

$$E^{c} = \frac{1}{L_0}\int_0^{L_0} |\alpha_{\text{crawl}}^{\text{TM}}|^2 \mathrm{d}s$$
$$E_i^{r}[n] = \frac{1}{L_0}\int_0^{L_0} \sum_{\hat{n}=1}^{\hat{f}_{\mathcal{W}_i}[n]} |\Delta\alpha_i[\hat{n}]|^2 \mathrm{d}s \tag{15}$$

where $\Delta\alpha_i[\hat{n}] = \max\{\alpha_i[\hat{n}] - \alpha_i[\hat{n}-1], 0\}$ is the difference between successive muscle activations. The maximum operator is used so that only actions leading to increased muscle activation are accounted for, i.e., there is no cost for relaxing muscles. If the low-level controller collects all food targets within reach (which is generally true if no obstacles are present) then $f_{\mathcal{W}_i}[n] = \hat{f}_{\mathcal{W}_i}[n]$.

We conclude by noting that the CyberOctopus problem described above bears similarities with the Traveling Salesman Problem (TSP).[55–57] However, unlike in the TSP, here the cost (muscle energy) required to reach the various food items is not known a priori and depends on both the CyberOctopus location and its arms' configurations, rendering optimization methods developed for the TSP not directly applicable.

4.3. Spatial Attention Heuristic and Reinforcement Learning Solution Method

In general, the high-level problem of Equation (13) has no analytical solution. This stems from the reward energy's dependence on the CyberOctopus' location and configuration which, as noted above, renders explicit planning methods (TSP) not amenable. We then resort to searching for an approximate solution, incorporating further insights from the octopus' behavior. Octopuses integrate visual, tactile, and chemosensory information to forage and hunt. However, in the wild, they are thought to primarily rely on visual cues.[58] For example, during foraging, they make behavioral decisions based on their distance from the den,[59] and they are able to discriminate between objects based on size, shape, brightness, and location.[60] These observations suggest the potential use of target prioritization strategies based on spatial location.

Adopting this insight, we define a cognitive attention heuristic wherein the CyberOctopus only pays attention to uncollected food targets that are within the attention space $\mathcal{V}$, ignoring all others. This attention space $\mathcal{V}$ can be flexibly defined depending on the task at hand (see Supporting Information for details). Here, we use an attention space that extends out twice the workspace distance (Figure 4a). The CyberOctopus' cognitive load can be further relieved by considering a fixed, maximum number of closest targets, allowing for immediate planning, while retaining sufficient environmental awareness for adequate longer-term decision-making. With this heuristic, the state of Equation (11) reduces to

$$z[n] = \{q_j[n]\}_{j=1}^{F} \tag{16}$$

where $\{q_j[n]\}_{j=1}^{F}$ are the positions (relative to the arm base) of the F closest uncollected food targets within $\mathcal{V}$. If fewer than F targets are within the attention range, the excluded entries are set to 0. If more than F targets are within the attention range, only the closest F are considered. This state definition is the one employed throughout the remainder of the article and is the one depicted in Figure 4b.

The use of this heuristic leads to a fixed state-space size, making the problem naturally amenable to reinforcement learning approaches. Here, we employ the proximal policy optimization (PPO) algorithm,[61] considered to be a state-of-the-art on-policy reinforcement learning scheme due to its robust performance. PPO utilizes an actor–critic architecture where an actor-network encodes the control policy, while a critic network estimates the quality of the control policy. Throughout the rest of the article, the control policy is encoded in a feedforward neural network with three hidden layers ($32 \times 32 \times 16$) of ReLU activation functions. The critic network shares the first two hidden layers with the control policy network but has a separate third hidden layer, also with 16 neurons.

5. A CyberOctopus Foraging for Food

Next, we put to the test the combined machinery described in Section 3 and 4 by simulating and characterizing a CyberOctopus foraging for food. To illustrate the use of primitives for high-level planning and reasoning, we first consider the reduced problem of a single active arm (Section 5.1), for which an analytical solution can be obtained under simplifying assumptions. After showing favorable comparison between learning-based and analytical solutions, in Section 5.2, we expand our approach to the case of multiple active arms (up to four), for which no analytical solutions exist. Finally, a third (lowest) level of control based on local physical compliance is incorporated, and a CyberOctopus outfitted with this trilevel hierarchy is shown to forage in an arena littered with obstacles in 3D space.

5.1. Foraging with One Arm

5.1.1. Analytical Solution

As mentioned above, there is no analytical solution to the high-level problem of Equation (13). However, under simplifying assumptions, analytical solutions may be obtained for the case of a single arm ($I = 1$). Here, we make the following assumptions: 1) The workspace $\mathcal{W}$ is treated as a rectangle whose width is larger than the distance the arm can crawl in one step; 2) The energetic cost of each high-level command is simplified to be a constant, with no dependence on the NN-ES muscle activations; 3) The food reward γ is greater than the constant cost to reach a target. Under these assumptions, an analytical Q-policy solution to the optimal dynamic programming (DP) planning problem of a CyberOctopus foraging with one arm can be derived, as detailed in the Supporting Information. We note that the same procedure can be extended to two arms crawling and reaching in two orthogonal planes (Supporting Information). However, if more than two arms are considered, the number of steps to reach all food targets can not be determined, making the derivation of an optimal analytical policy not possible, even under the above simplifying assumptions. Despite their limited scope, one- and two-arm analytical solutions are still useful to benchmark and contextualize our hierarchical approach and its learning-based

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

solution, as we progress to the more general, nonanalytically tractable scenario of larger numbers of engaged arms.

5.1.2. Reinforcement Learning Solution

We proceed with solving the problem of Equation (11)–(16) for one arm, via the PPO reinforcement learning algorithm, using the setup described in Section 4.3. The performance of the one-arm policy, numerically obtained with PPO and dynamically executed by the CyberOctopus in Elastica, is shown in **Figure 5**a, while policy convergence during training is illustrated in Figure 5b. The policy is trained for 2000 epochs, each epoch entailing 1024-time steps. At the beginning of each episode, 20 targets are randomly generated in a rectangular area within the arm's bending plane (Figure 1b), and the arm is initialized in a straight configuration on the left side of the food (Figure 5a). An episode terminates when either all targets have been reached or the number of time steps exceeds 180, so that each epoch contains at least five episodes.

The CyberOctopus (whose only active arm is depicted in blue) successfully learns to crawl toward food, positioning itself for reaching until all food in the environment is collected (Figure 5a and Video S1, Supporting Information). To contextualize this performance, we implement three alternative high-level controllers: the simplified, analytical Q-policy solution described earlier, a "greedy" controller that immediately reaches for food whenever available, otherwise, it chooses to crawl, and a random controller that selects crawl or reach all with equal probability at each time step. We evaluate the performance of these four controllers on three metrics (Figure 5c): 1) net energy (Equation (13)), 2) average number of time steps per food item collected, and 3) average number of crawl steps per food target. We find, unsurprisingly, that the random policy performs the worst, utilizing, on average, 25% more energy and 40% more steps per episode than the other approaches. The greedy and Q-policy controllers perform comparably, though the Q-policy presents a wider distribution than greedy, likely due to simplifying assumptions that cause the occasional miss of a target. Overall, the PPO policy exhibits the best performance, strategically crawling until a large number of food items are simultaneously within reach, to then fetch them all at once. In light of the energy cost maps of Figure 3d, this learned approach intuitively correlates to lower muscle activation costs. This strategy (which is also adopted by the Q-policy in the simplified setting of the problem) not only reduces energy costs but additionally allows completion of the foraging task in a fewer number of time steps (Figure 5c). As we will see, the performance gap between PPO and the alternative approaches will only widen as more complex scenarios ($I > 1$) are considered, until becoming the only reliable option.

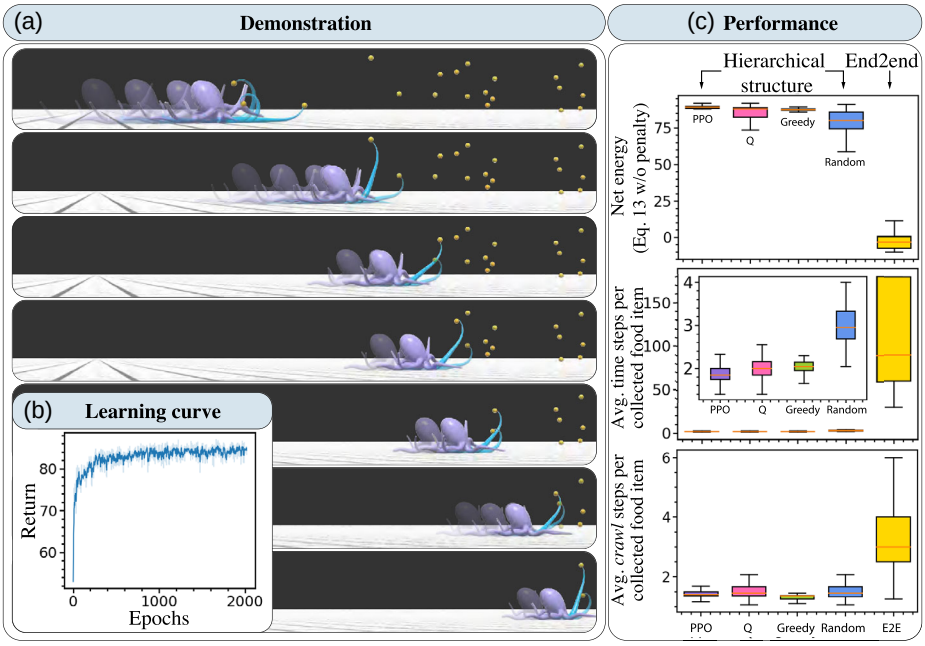

Figure 5. Hierarchical control framework. Based on the locations (relative to the arm base) of targets in the view range, the high-level controller decides between two actions, *crawl* or *reach all*, to maximize gained energy. a) Demonstration of learned PPO policy for a single active arm as implemented in Elastica. The active arm is depicted in blue, with the levels of shading indicating intermediate configurations. The CyberOctopus successfully moves forward and collects available food. Here, $F = 5$, $\gamma = 5$, and $\Phi = 1.0$. A video of this demonstration is available (Video S1, Supporting Information). b) Learning curve of the PPO algorithm for a single active arm. c) Key metrics for evaluating the performance of hierarchical and end-to-end (e2e) control schemes. Orange lines represent the metric's median value, boxes represent the interquartile (middle 50%) range, and whiskers denote the min and max of 100 evaluation samples. PPO is the reinforcement learning algorithm, Q is the analytic solution of a simplified DP problem (see Supporting Information for details), the greedy policy chooses to reach anytime there is food within reach, else it crawls, and the random policy chooses actions randomly. The e2e approach attempts to directly solve the end-to-end problem (see Supporting Information for details). Hierarchy-based control policies are found to outperform end-to-end solutions, with PPO performing best overall.

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

To comparatively isolate the benefits provided by our hierarchical decomposition, we solve the same foraging problem in an end-to-end (e2e) fashion, using PPO. In other words, we train a single network to directly map food locations and current muscle activations to output muscle activations, completely bypassing the decomposition between high-level planning and low-level execution. A comparison between the two approaches is schematically illustrated through the block diagrams of Figure 4b,c, with further mathematical and implementation details available in the Supporting Information. We note that while the previous activation $\alpha[\hat{n}-1]$ is used by the low-level controller in Figure 4b, in the end-to-end formulation depicted in Figure 4c, this information is included in the state representation. Thus, overall, the same amount of information is provided to both frameworks.

We find that all four hierarchical policies (including the random policy) outperform the end-to-end approach, across all metrics (Figure 5c), despite significant effort in tuning training parameters. This result underscores how the separation of concerns enabled by the hierarchy significantly simplifies control, and thus learning. Further, it illustrates the potential of a mixed-modes approach, where model-free and model-based algorithms are employed at different levels so as to synergize and complement each other.

5.2. Foraging with Multiple Arms

We next gradually increase the number of active arms engaged by the CyberOctopus, considering the first two arms and then four arms. As the number of arms increases, the CyberOctopus is able to access additional directions of movement and must coordinate its arm's behaviors, significantly increasing the problem difficulty. Additionally, in the four-arm case, we incorporate obstacles into the arena, requiring the arms to exploit their mechanical intelligence to avoid becoming obstructed.

5.2.1. Foraging with Two Arms

We first consider a CyberOctopus with two active arms ($I = 2$), orthogonal to each other. Training proceeds as with the single arm, although now 10 000 epochs are used. For each episode, the arms are initialized at rest and 20 food locations are scattered randomly throughout the arena on vertical planes that align with the grid formed by discrete crawling steps. As reported in **Figure 6**, the learning-based approach successfully converges (see Supporting Information for learning curves), as reflected in the evaluation of the corresponding policy in Figure 6b. The behavior learned is illustrated in Figure 6a as well as in Video S2, Supporting Information. As can be seen, the CyberOctopus crawls between targets and collects them, with the learned policy successfully coordinating its two independent arms so as to crawl and switch along orthogonal directions, simultaneously grabbing food with both arms or fetching food with one arm while crawling with the other arm. We again contextualize our results by means of three alternative high-level controllers: Q is the analytically identified solution of the simplified DP problem described above and in the Supporting Information, greedy has each arm collecting food when immediately available, else crawling (Supporting Information for details), and random selects *crawl* or *reach all* with equal probability for each arm. Again the learning-based solution outperforms the alternative high-level controllers, consistently collecting more energy and using fewer steps to do so (Figure 6b). We forego a comparison with the end-to-end approach, as its performance is deemed too poor to provide any useful insight. This once again underscores the significant

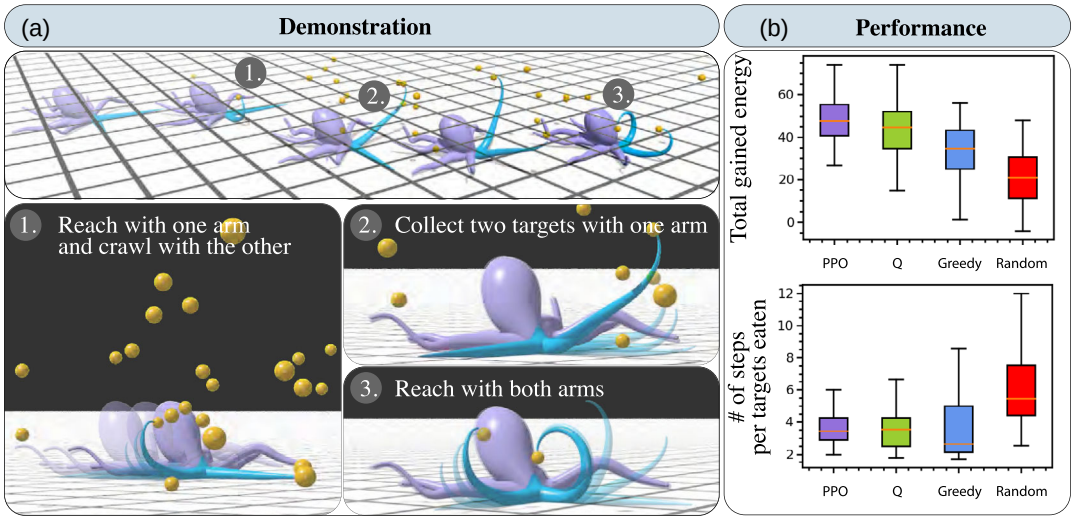

Figure 6. a) Demonstration in Elastica of learning-based PPO policy controlling two arms to forage. (a1) The arms coordinate their actions to move across two dimensions and reach targets. (a2) One arm collects multiple targets within its bending plane. (a3) Two arms simultaneously collect targets in their respective bending planes. A video of this demonstration is available (Video S2, Supporting Information). b) Performance of the different control schemes. Orange lines represent the metric's median value, boxes represent the interquartile (middle 50%) range, and whiskers denote the min and max of 100 evaluation samples. The learning-based PPO policy gains more total energy and requires fewer steps per collected target than the alternative high-level policies. For the learning-based policy, $F = 10$. Learning curves are reported in the Supporting Information.

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

impact of hierarchically decomposing the problem, which in this example amounts to being able to solve the problem versus not being able to do so (end-to-end).

5.2.2. Foraging with Four Arms and in the Presence of Solid Obstacles

Having established the viability of our multiarm approach, we now consider the problem of a CyberOctopus foraging with four active arms ($I = 4$). The arms, being orthogonal to each other, allow the CyberOctopus to fully traverse the substrate by crawling along the four cardinal directions, rendering the foraging problem analytically intractable. Further, this time, we consider the additional scenario in which not only food but also solid obstacles are distributed in 3D space (**Figure 7**a). The goal is twofold: first, we wish to characterize the ability of our methods to learn to forage without obstacles, and second, we wish to explore how principles of mechanical intelligence[31,62] may be used to deal with environmental disturbances (obstacles), without further burdening control or training.

To this end, we incorporate a simple behavioral reflex based on traveling waves of muscle activation observed in the octopus.[47,48] When contact between an arm and an obstacle is sensed, two waves emanate from the point of contact in each direction. One, traveling toward the arm's tip, signals all muscles to relax, while the second, traveling toward the arm's base, signals longitudinal muscles on the contacting side to increase activation with all other muscles relaxing. Here, we treat these waves as propagating instantaneously. Once contact ceases, the arm returns to executing the originally prescribed muscle activations. As can be seen in Figure 7a, this reflex, mediated by the compliant nature of the arm, allows the latter to slip past obstacles, thus dealing with their presence with minimal additional computational effort. Instead, without reflex, the arm routinely gets stuck (Figure 7c).

The four-arm CyberOctopus is then initialized in the center of the arena with 40 food items randomly distributed in 3D space. Training proceeds *without* considering obstacles, using the same process of the two-arm case, and the CyberOctopus successfully learns to forage (Figure 7b, learning curves in the Supporting Information). In contrast to the two-arm case, here the CyberOctopus' four active arms enable forward/backward and left/right crawling, allowing the collection of previously missed food, at a later stage. This additional freedom increases the planning required to efficiently move throughout the area, not only making an analytical Q-policy impossible to define, but also substantially impairing the ability of the greedy policy to collect food. The greedy policy (see Supporting Information for details) is indeed found to be able to collect only 38% of the available food, a notable decrease from the 54% of food collected with two arms. As for the random policy (and end-to-end), we forgo a comparison due to the too poor performance. In contrast, the PPO learning-based approach is able to successfully exploit this additional freedom (four cardinal directions of motion) to improve food collection, fetching on average 88% of the food compared to 66% when only two arms are active (Figure 7c). The learning-based policy (trained *without* obstacles) is then deployed in an environment littered with unmovable obstacles leading to arms becoming stuck, degrading food collection, which now only achieves 21%. With the sensory reflex enabled, however, the CyberOctopus successfully recovers. Figure 7c shows the CyberOctopus utilizing this reflexive behavior to reach previously

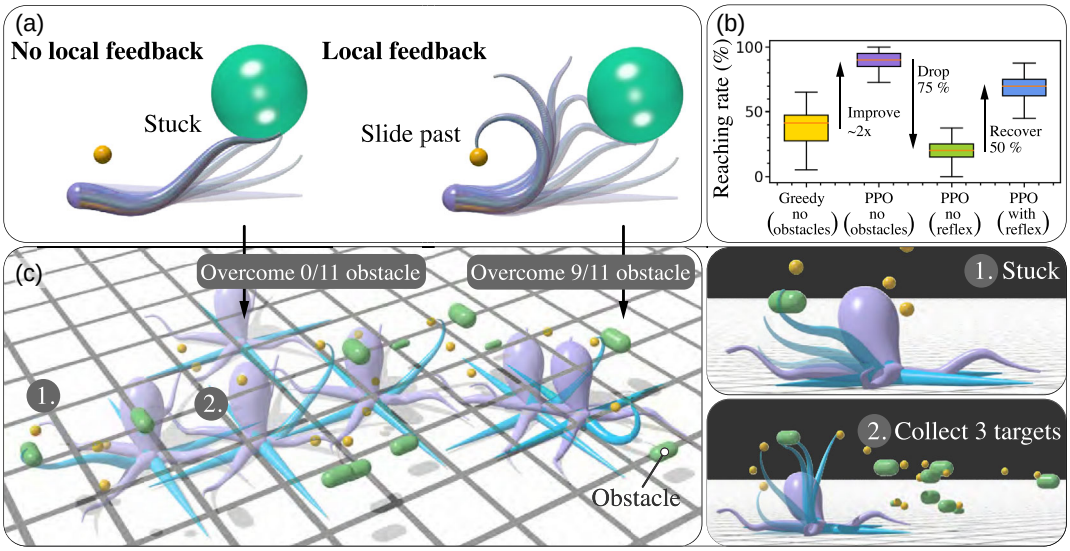

Figure 7. a) Demonstration of an arm sliding past an obstacle when equipped with local sensory feedback. b) Percentage of available food items collected from the environment by the greedy policy (yellow) and the learning-based policy when no obstacles are present (purple), when the arm does not utilize a reflexive behavior (green), and when the reflex is engaged (blue). Orange lines represent the median performance over 100 episodes, boxes represent the interquartile (middle 50%) range, and whiskers denote the min and max. c) Demonstration of a foraging CyberOctopus with four active arms (blue). Using the local sensory reflex allows the arms to overcome 9 of the 11 obstacles that otherwise would cause the arm to become stuck. (c1) The arms coordinate their actions to move across two dimensions and reach targets. (c2) One arm collects multiple targets within its bending plane. A video of this demonstration is available (Video S3, Supporting Information). Learning curves are reported in the Supporting Information.

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

obstructed food (Video S3, Supporting Information), resulting in 67% of food collected.

6. Conclusion

Recognizing the need for improved control methods in multiarm soft robots, we propose a hierarchical framework inspired by the organization of the octopus neurophysiology and demonstrate it in a CyberOctopus foraging for food. By decomposing control into high-level decision-making, low-level motor activation, and reflexive modulation via local sensory feedback and mechanical compliance, we show significant improvements relative to end-to-end approaches. Performance is enabled via a mixed-modes approach, whereby complementary control schemes can be swapped out at any level of the hierarchy. Here, we combine model-free reinforcement learning, for high-level decision-making, and model-based energy shaping control, for low-level muscle recruitment and activation. To enable compatibility in terms of computational costs, we developed a novel, NN-ES controller that accurately executes arm motor programs, such as reaching for food or crawling, while exhibiting time-to-solutions more than 200x faster than previous attempts.[25] Our hierarchical approach is successfully deployed in progressively challenging foraging scenarios, entailing two- and three-dimensional settings, solid obstacles, and increasing number of active arms.

Overall, this work presents a framework to explore the control of multiple, compliant, and distributed arms, in both engineering and biological settings, with the latter providing insights and hypotheses for computational corroboration. We have begun to take initial steps in this regard, testing how principles of mechanical intelligence and muscle waves of activation[47,48] may be couched into local reflexive schemes for accommodating solid obstacles. Future work will build on these foundations to include three-dimensional dynamics and further explore biologically plausible forms of distributed control.[31]

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

The authors gratefully acknowledge Tigran Norekian and Ekaterina D. Gribkova who performed the histology of the octopus arm in Figure 2a. This study was jointly funded by ONR MURI N00014-19-1-2373 (M.G., P.G.M.), ONR N00014-22-1-2569 (M.G.), NSF EFRI C3 SoRo #1830881 (M.G.), and with computational support provided by the Bridges2 supercomputer at the Pittsburgh Supercomputing Center through allocation TG-MCB190004 from the Extreme Science and Engineering Discovery Environment (XSEDE; NSF grant ACI-1548562).

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The open-source Python implementation of Elastica is available at www.github.com/GazzolaLab/PyElastica. All codes used in this paper are available at https://github.com/chshih2/Real-time-control-of-an-octopus-arm-NNES and https://github.com/chshih2/Multi-arm-coordination-for-foraging.

Keywords

bioinspiration, hierarchical control, soft robotics

Received: February 11, 2023
Revised: April 30, 2023
Published online:

[1] A. Sadeghi, A. Mondini, B. Mazzolai, *Soft Rob.* **2017**, *4*, 211.
[2] X. Zhang, F. Chan, T. Parthasarathy, M. Gazzola, *Nat. Commun.* **2019**, *10*, 4825.
[3] H. Huang, F. Uslu, P. Katsamba, E. Lauga, M. Sakar, B. Nelson, *Sci. Adv.* **2019**, *5*, eaau1532.
[4] C. Della Santina, C. Duriez, D. Rus, arXiv preprint arXiv:2110.01358 **2021**, 1–69.
[5] T. George Thuruthel, Y. Ansari, E. Falotico, C. Laschi, *Soft Rob.* **2018**, *5*, 149.
[6] D. Trivedi, C. Rahn, W. Kier, I. Walker, *Appl. Bionics Biomech.* **2008**, *5*, 99.
[7] M. W. Hannan, I. D. Walker, *Adv. Rob.* **2001**, *15*, 847.
[8] M. Baumgartner, F. Hartmann, M. Drack, D. Preninger, D. Wirthl, R. Gerstmayr, L. Lehner, G. Mao, R. Pruckner, S. Demchyshyn, L. Reiter, M. Strobel, T. Stockinger, D. Schiller, S. Kimeswenger, F. Greibich, G. Buchberger, E. Bradt, S. Hild, S. Bauer, M. Kaltenbrunner, *Nat. Mater.* **2020**, *19*, 1102.
[9] A. C. Noel, D. L. Hu, *J. Exp. Biol.* **2018**, *221*, jeb176289.
[10] X. Lu, W. Xu, X. Li, in *2015 6th Int. Conf. on Automation, Robotics and Applications (ICARA)*, Queenstown, New Zealand **2015**, pp. 332–336, https://doi.org/10.1109/ICARA.2015.7081169.
[11] E. B. Joyee, Y. Pan, *Soft Rob.* **2019**, *6*, 333.
[12] B. Gamus, L. Salem, A. D. Gat, Y. Or, *IEEE Rob. Autom. Lett.* **2020**, *5*, 1397.
[13] P. Karipoth, A. Christou, A. Pullanchiyodan, R. Dahiya, *Adv. Intell. Syst.* **2022**, *4*, 2100092.
[14] I. Must, E. Sinibaldi, B. Mazzolai, *Nat. Commun.* **2019**, *10*, 344.
[15] J. M. Skotheim, L. Mahadevan, *Science* **2005**, *308*, 1308.
[16] M. Otake, Y. Kagami, M. Inaba, H. Inoue, *Rob. Auton. Syst.* **2002**, *40*, 185.
[17] H. Jin, E. Dong, G. Alici, S. Mao, X. Min, C. Liu, K. Low, J. Yang, *Bioinspiration biomimetics* **2016**, *11*, 056012.
[18] X. Zhang, N. Naughton, T. Parthasarathy, M. Gazzola, *Nat. Commun.* **2021**, *12*, 6076.
[19] H. Marvi, C. Gong, N. Gravish, H. Astley, M. Travers, J. Hatton, R. L. Mendelson, H. Choset, D. Hu, D. Goldman, *Science* **2014**, *346*, 224.
[20] T. Li, K. Nakajima, M. Calisti, C. Laschi, R. Pfeifer, *2013 IEEE Int. Conf. on Mechatronics and Automation*, Karlsruhe, Germany, **2013**, 1504–1511, https://doi.org/10.1109/ICRA.2013.6630770.
[21] M. Cianchetti, M. Calisti, L. Margheri, M. Kuba, C. Laschi, *Bioinspiration Biomimetics* **2015**, *10*, 3.
[22] M. D. Grissom, V. Chitrakaran, D. Dienno, M. Csencits, M. Pritts, B. Jones, W. McMahan, D. Dawson, C. Rahn, I. Walker, *Unmanned Systems Technology VIII*, Vol. *6230*, May 2006, p. 62301F.

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

[23] Q. Wu, X. Yang, Y. Wu, Z. Zhou, J. Wang, B. Zhang, Y. Luo, S. A. Chepinskiy, A. A. Zhilenkov, *Bioinspiration Biomimetics* **2021**, *16*, 046007.
[24] Y. Sakuhara, H. Shimizu, K. Ito, in *2020 IEEE 10th Int. Conf. on Intelligent Systems (IS)*, Varna, Bulgaria **2020**, pp. 463–468, https://doi.org/10.1109/IS48319.2020.9199967.
[25] H.-S. Chang, U. Halder, E. Gribkova, A. Tekinalp, N. Naughton, M. Gazzola, P. G. Mehta, In *2021 60th IEEE Conference on Decision and Control (CDC)*, Austin, TX **2021** pp. 1383–1390, https://doi.org/10.1109/CDC45484.2021.9683318.
[26] J. Walker, C. Ghalambor, O. Griset, D. McKenney, D. Reznick, *Funct. Ecol.* **2005**, *19*, 808.
[27] F. Grasso, *Am. Malacol. Bull.* **2008**, *24*, 13.
[28] E. Kennedy, K. C. Buresch, P. Boinapally, R. T. Hanlon, *Sci. Rep.* **2020**, *10*, 20872.
[29] P. Godfrey-Smith, *Sci. Am. Mind* **2017**, *28*, 62.
[30] J. Young, *J. Anat.* **1971**, *112*, 144.
[31] B. Hochner, *Curr. Biol.* **2012**, *22*, R887.
[32] G. Sumbre, Y. Gutfreund, G. Fiorito, T. Flash, B. Hochner, *Science* **2001**, *293*, 1845.
[33] T. Hague, M. Florini, P. L. Andrews, *J. Exp. Mar. Biol. Ecol.* **2013**, *447*, 100.
[34] T. Gutnick, L. Zullo, B. Hochner, M. J. Kuba, *Curr. Biol.* **2020**, *30*, 4322.
[35] H.-S. Chang, U. Halder, C.-H. Shih, A. Tekinalp, T. Parthasarathy, E. Gribkova, G. Chowdhary, R. Gillette, M. Gazzola, P. G. Mehta, in *2020 59th IEEE Conf. on Decision and Control (CDC)*, Jeju, Korea (South), **2020**, pp. 3913–3920, https://doi.org/10.1109/CDC42340.2020.9304408.
[36] M. Gazzola, L. H. Dudte, A. G. McCormick, L. Mahadevan, *R. Soc. Open Sci.* **2018**, *5*, 6.
[37] W. Kier, M. Stella, *J. Morphol.* **2007**, *268*, 831.
[38] A. Tekinalp, N. Naughton, S.-H. Kim, U. Halder, R. Gillette, P. G. Mehta, W. Kier, M. Gazzola, arXiv preprint arXiv:2304.08413 **2023**.
[39] S. S. Antman, *Nonlinear Problems of Elasticity*, Springer, Berlin **1995**.
[40] B. Zhang, Y. Fan, P. Yang, T. Cao, H. Liao, *Soft Rob.* **2019**, *6*, 399.
[41] N. Naughton, J. Sun, A. Tekinalp, T. Parthasarathy, G. Chowdhary, M. Gazzola, *IEEE Rob. Autom. Lett.* **2021**, *6*, 3389.
[42] O. Aydin, X. Zhang, S. Nuethong, G. Pagan-Diaz, R. Bashir, M. Gazzola, M. Saif, *Proc. Natl. Acad. Sci.* **2019**, *116*, 19841.
[43] G. Pagan-Diaz, X. Zhang, L. Grant, Y. Kim, O. Aydin, C. Cvetkovic, E. Ko, E. Solomon, J. Hollis, H. Kong, T. Saif, M. Gazzola, R. Bashir, *Adv. Funct. Mater.* **2018**, *28*, 1801145.
[44] J. Wang, X. Zhang, J. Park, I. Park, E. Kilicarslan, Y. Kim, Z. Dou, R. Bashir, M. Gazzola, *Advanced Intelligent Systems* **2021**, *3*, 2000237.
[45] N. Charles, M. Gazzola, L. Mahadevan, *Phys. Rev. Lett.* **2019**, *123*, 208003.
[46] F. Tramacere, A. Kovalev, T. Kleinteich, S. N. Gorb, B. Mazzolai, *J. R. Soc. Interface* **2014**, *11*, 20130816.
[47] Y. Gutfreund, T. Flash, G. Fiorito, B. Hochner, *J. Neurosci.* **1998**, *18*, 5976.
[48] G. Sumbre, G. Fiorito, T. Flash, B. Hochner, *Curr. Biol.* **2006**, *16*, 767.
[49] H.-S. Chang, U. Halder, C.-H. Shih, N. Naughton, M. Gazzola, P. G. Mehta, *Proc. R. Soc. A* **2023**, *479*, 20220593.
[50] R. Ortega, A. J. Van Der Schaft, I. Mareels, B. Maschke, *IEEE Control Syst. Mag.* **2001**, *21*, 18.
[51] G. Blankenstein, R. Ortega, A. J. Van Der Schaft, *Int. J. Control* **2002**, *75*, 645.
[52] A. M. Bloch, N. E. Leonard, J. E. Marsden, *IEEE Trans. Autom. Control* **2000**, *45*, 2253.
[53] G. Levy, N. Nesher, L. Zullo, B. Hochner, in *The Oxford Handbook Of Invertebrate Neurobiology*, Oxford University Press Oxford **2019**, ISBN 9780190456757.
[54] F. Bidel, N. C. Bennett, T. J. Wardill, *Current Biology* **2022**, *32*, 4780.
[55] D. L. Applegate, R. E. Bixby, V. Chvátal, W. J. Cook, in *The Traveling Salesman Problem*. Princeton University Press, Princeton **2011**.
[56] G. Gutin, A. P. Punnen, *The Traveling Salesman Problem and its Variations*, vol. *12*, Springer Science & Business Media, Berlin **2006**.
[57] C. H. Papadimitriou, *Theor. Comput. Sci.* **1977**, *4*, 237.
[58] V. Maselli, A.-S. Al-Soudy, M. Buglione, M. Aria, G. Polese, A. Di Cosmo, *Animals* **2020**, *10*, 457.
[59] J. A. Mather, *J. Comp. Physiol. A* **1991**, *168*, 491.
[60] J. Boal, *Biol. Rev. Cambridge Philos. Soc.* **1996**, *71*, 157.
[61] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, arXiv preprint arXiv:1707.06347 **2017**, pp. 1–12.
[62] G. Mengaldo, F. Renda, S. L. Brunton, M. Bächer, M. Calisti, C. Duriez, G. S. Chirikjian, C. Laschi, *Nat. Rev. Phys.* **2022**, *4*, 595.

© 2023 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH

26404567, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/aisy.202300088 by Readcube (Labtiva Inc.), Wiley Online Library on [23/06/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License