2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)
April 4-8, 2022, Edinburgh, UK

Strain-Based Consensus in Soft, Inflatable Robots

Alexandra Nilles[1], Steven Ceron[2], Nils Napp[1], and Kirstin Petersen[1]

***Abstract*— Soft robots actuate themselves and their world through induced pressure and strain, and can often sense these quantities as well. We hypothesize that coordination in a tightly coupled collective of soft robots can be achieved with purely *proprioceptive* sensing and no direct communication. In this paper, we target a platform of soft pneumatic modules capable of sensing strain on their perimeter, with the goal of using only the robots' own soft actuators and sensors as a medium for distributed coordination. However, methods for modelling, sensing, and controlling strain in such soft robot collectives are not well understood. To address this challenge, we introduce and validate a computationally efficient spring-based model for two-dimensional sheets of soft pneumatic robots. We then translate a classical consensus algorithm to use only proprioceptive data, test in simulation, and show that due to the physical coupling between robots we can achieve consensus-like coordination. We discuss the unique challenges of strain sensors and next steps to bringing these findings to hardware. These findings have promising potential for smart materials and large-scale collectives, because they omit the need for additional communication infrastructure to support coordination.**

I. INTRODUCTION

Soft robotics has rapidly developed over the last two decades, demonstrating simple mechanisms with infinite degrees of freedom and the ability to passively deform to external objects [1]. Recently, this field has started to impact swarm robotics, where many simple robots coordinate locally to achieve complex emergent behaviors [2]. The two fields complement each other well. Where soft robots have limited active degrees of freedom and sensory capabilities, swarms of soft robots can pool resources to achieve capabilities beyond the sum of the parts. Meanwhile, soft swarms have scalable, affordable, multi-functional hardware that is robust to physical interactions between robots and the environment.

In this article we are inspired by epithelial tissues that primarily use inter-cell forces to regulate shape and structure and generate signals that influence tissue growth [3]. This method of coordination through strain in the shared substrate is uniquely suited to collectives of soft robots because they intrinsically couple strain, actuation, and sensing. This lets us add a layer of system intelligence embodied in the information-rich interaction between deformable agents.

Adapting methods from distributed robotics to strict proprioceptive sensing requires careful design of both hardware and algorithms. Traditional consensus algorithms require agents to directly observe their neighbor's state or communicate between agents, and often assume that an individual's state change does not affect its neighbors. In contrast, strain is a proprioceptive sensing technique that does not necessarily allow for full reconstruction of neighboring robot states. Furthermore, in coupled soft robot collectives, the change of state (strain) in one individual will have a direct impact on the state of the neighbors. Our work examines how forces couple in the collective, and shows that individually felt strain can serve as a proxy for direct observation of the local state of the collective. To support our study, we leverage the latest iteration of the "Foambots", where many magnetically coupled pneumatic modules can induce and measure strain in the collective [4]. We develop a low-dimensional, computationally efficient model and a corresponding simulator, fitted to and validated against the hardware. We then use this simulator to investigate how to adapt distributed consensus to a setting where modules cannot directly communicate, but only change their own pneumatic state in response to their own strain measurements. Our work suggests that if robotic agents comprise the substrate they are actuating, strain sensing can be an effective method for low-bandwidth distributed coordination, and these insights can be of benefit to both soft modular robots and smart materials [5].

Our contributions are the design and characterization of an updated soft pneumatic robot module (Sec. III); a simple model of static forces in a collective of such modules (Sec. IV); the calibration and validation of a software implementation of the model (Sec. IV-C); and finally, implementation of distributed consensus algorithms in simulation and a comparison of performance under different constraints on communication and sensing (Sec. V).

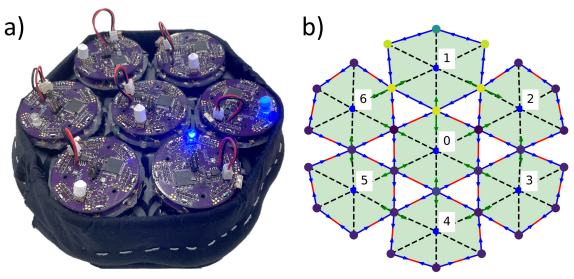


Fig. 1: Platform used to study consensus in soft robot collectives. (a) Seven foambots in a dense packing, inside a compliant fabric sleeve. (b) Seven simulated robots (green polygons) in a dense configuration, showing simulated strain sensor responses (nondimensional).

This work was supported by National Science Foundation grants DMR-1933284, CNS-2042411, IIS-1846340, and a Packard Fellowship for Science and Engineering.

[1]School of Electrical and Computer Engineering, Cornell University, Ithaca, NY 14853, USA. [2]Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14853, USA. aqn3@cornell.edu, sc2775@cornell.edu, nnapp@cornell.edu, kirstin@cornell.edu

2022 IEEE 5th International Conference on Soft Robotics (RoboSoft) | 978-1-6654-0828-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/ROBOSOFT54090.2022.9762180

II. Related Work

Several past works have suggested ways in which decentralized coordination can be achieved in soft/deformable robot collectives. In [6], deformable robots composed of flexible PCBs used electromagnetic communication to pass explicit messages between coupled magnets on their perimeters. In the ant-inspired robots in [7] and the voice-coil-based robots in [8], the authors leveraged mechanical/acoustical vibration to pass explicit messages between robots. Other works have leveraged physical interactions, rather than robot-robot communication, between non-flexible robotic oscillators to achieve behaviors such as collective motion by understanding the statistical mechanics of the systems [9], [10]. Also related is coordination through robot-medium interactions, such as a constrained collective of mobile robots in granular media [11], where the interactions between robots and their substrates allow the collective to switch between several useful modes and thus achieve both locomotion and grasping behaviors. In contrast to these systems, the focus of this paper is to demonstrate how collective coordination can be achieved through distributed sensing and actuation of strain, which is often implicit to the functionality of soft robots. Where others mostly focus on producing collective motion, we focus on strain consensus, though our approach may be extended to motion in future work.

Other works have investigated the effect of adding a confining boundary or enclosure to collections of very simple robots [12], [13], and it is a branch of research that is very promising for micro-scale applications. There remain many open questions about the effects of scale, material properties, and geometry of the boundary on collective behavior in confinement. In our work, the compliant perimeter is used to provide more realistic data for simulator validation, but future work will investigate more direct use of enclosures and robot-environment interactions.

Best approaches to modelling soft robots is an area of active research, with methods including rod and beam models, piece-wise constant curvature models, and finite-element approaches [14]. In addition, model-free approaches have been well-established in similar modular, soft systems as long as they are sufficiently observable [15]. Since our robots are limited to relatively slow actuation, we do not consider dynamic effects, and we target distributed algorithms that only use each agent's proprioceptive sensors. We support this work by developing an efficient, validated quasi-static simulator and use it to analyze said algorithms.

III. HARDWARE PLATFORM

The experimental results are obtained using the Foambots, originally described in [4]. In this paper we introduce and use a newer version of these soft modular oscillators, shown in Fig. 2(a-b). Modules are cylindrical, measuring 84mm in height with a diameter of 54mm. Their perimeter consists of an inflatable EcoFlex™ 50 membrane with integrated: 1) spandex fabric for resilience, 2) six pairs of embedded Neodymium magnets (B631 from K&Jmagnetics) for loose coupling with their neighbors, and 3) six embedded strain sensors fabricated as described in [16] to perceive forces acting on their membrane. Modules have a rigid, 3D printed core which holds a DIMINUS 6V air pump, an Uxcell 3mm normally-closed valve, and a 3.7V 600mAh Li-Ion battery for stand-alone operation. Finally, a printed circuit board on top of the cylinder contains an ATmega324 8bit microcontroller, six infrared transceivers, a NXP absolute pressure sensor (max 16.7PSI), and an RGB debugging LED.

A module is capable of increasing its circumference by $\sim$1.85 times, from 54mm diameter to 100mm diameter. In spite of the high strain ratio, the membrane response is close to linear (Fig. 2(c)), likely due to the embedded latex fabric. It is worth noting that strain sensors capable of this level of extension are typically based on liquid metals which risk leakage [1]. Although the resistive sensors on the Foambots maintain conductivity at maximum strain, they also exhibit non-linear, temporal behaviors (Fig. 2(d), detailed in [16]).

In Fig. 2(e), we show how the spring constant of a module and its membrane changes at different inflation states. To measure the spring constant of a module, we placed a single inflated module between a hard wall and a 5kg micro load cell from Phidgets, and varied the distance from the load cell to the wall, so that the load cell compressed the module. To measure the spring constant of the membrane, we fixed one end of membrane material and attached the load cell to the other end and stretched it. From this experiment we see 1) that force, F, and displacement, x, have a linear relationship in all tests; 2) the membrane has a lower spring constant, k_m, than the internal spring constant, k_i, of the inflated modules; and 3) when the module is inflated more (as $t_{inflation}$ increases), it leads to a higher internal spring constant. It is also worth noting that at $t_{inflation} > 7s$, the membrane reaches its maximum possible strain. At this point the pressure inside the Foambot increases markedly and it becomes much stiffer. Excluding this data point, we see that the Foambot spring constant has a linear relationship with the amount of inflation.

IV. MODEL AND SIMULATION

We present a simple model that predicts the steady-state configuration of our robot collective, given inflation setpoints for each robot and the contact topology of the robot collective. By modelling key aspects of the collective as a network of springs, we can efficiently estimate how forces in the collective will scale and change during state changes in the collective. We are also able to run simulations with larger numbers of agents, as a first step toward understanding behaviors of large collectives of deformable agents.

In simulation, a Foambot is constructed from six *internal* springs representing the internal air pressure of the Foambot and six *membrane* springs representing the membrane. The spring constants for the simulator were chosen to reflect the properties of the physical Foambots as described in Sec.III. The six nodes on the perimeter of each robot represent the six sensors, and if agents are attached, they are attached at a sensor node. We are interested primarily with how forces and strain propagate through the system, especially their relative

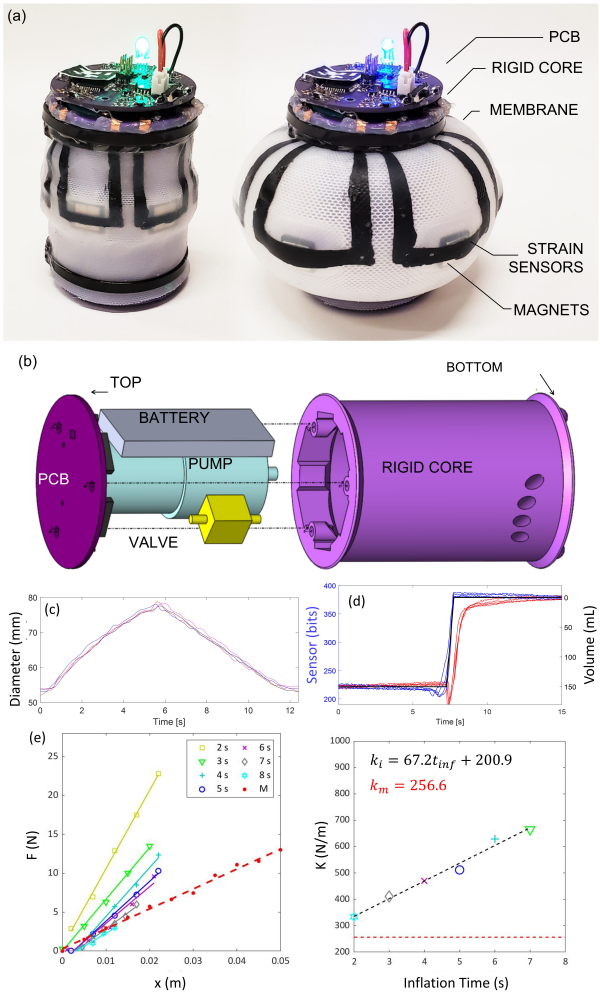


Fig. 2: (a) Foambot module in deflated and inflated state. (b) Exploded view showing rigid core design. (c) Membrane diameter over 4 inflation and deflation cycles. (d) Hysteresis in sensor response to rapid inflation and deflation. (e-left) Force versus displacement of membrane/module inflated for [2, 3.. 8]s. (e-right) Spring constants.

magnitudes as measured by the sensors on individual robots. The Foambots inflate and deflate slow, therefore we assume a quasi-static system which allows us to use least squares optimization to compute the expected system configuration and forces after the robots perform their actuation. This low-dimensional model is simple enough to be useful for quick iteration on algorithms and physical designs.

To compute the equilibrium position of the system, we use a least-squares formulation. The input variables are the xy-coordinates of the vertices and each spring has an associated output variable with the value $\sqrt{k}(l - l_0)$, where k and l_0 are properties of the spring and the actual length l is computed from the input variables. Physically, this means the least squares solution corresponds to a minimum energy configuration. This cost function is invariant to isometric transformations (translation and orientation), and to aid visualization we fix the orientation and location by excluding one "anchor" point from the input variables and adding an additional output variable that corresponds to the orientation of an "anchor edge".

To compute the solution we use SciPy's built in least squares solver with the default Trust Region Reflective (TRF) algorithm. Since each node is only connected to a few springs, the TRF solver can take advantage of this inherent sparsity. To utilize this feature, we explicitly compute the Jacobian as a sparse matrix (`csr_sparse`). This provides a speed up of several orders of magnitude compared to the default 2-point method for computing the Jacobian and allows us to solve a system of several thousand springs (hundreds of robots) in a few seconds.

In situations where we want to model a strain-limiting boundary, we compute the overall boundary perimeter, and once it reaches a maximum length P, add a new set of stiff springs to the edges on the boundary that enforce this constraint. When the boundary is in tension and shrinks beyond the P threshold we remove these added edges. The source code of the simulator is available upon request.

A. Simulating Strain Sensors

The strain sensors used in the Foambots generally increase in resistance when they are stretched or compressed, so the sensor signal is a function of both the perimeter geometry of the robot and any external forces acting on the membrane. Following work in modelling of biological epithelial systems [17], we model the strain-inducing forces acting on sensor vertex j of robot α with neighboring robot β as

$$F(v_j^\alpha) = k_m \Bigg(\sum_{e \in CCW} \sigma(e)(l_e^\alpha - l_e^{\alpha,rest})\hat{\boldsymbol{u}}_j^\alpha + \sum_{f \in CW} \sigma(f)(l_f^\alpha - l_f^{\alpha,rest})\hat{\boldsymbol{v}}_j^\alpha \Bigg) + A\Big(k_i^\beta(\ell_j^\beta - \ell_j^{\beta,rest})\hat{\boldsymbol{l}}_j^\beta + k_i^\alpha(l_j^\alpha - l_j^{\alpha,rest})\hat{\boldsymbol{l}}_j^\alpha \Big). \tag{1}$$

Fig. 3 illustrates the vectors involved in the force model at a single sensor. At a high level, the first term quantifies strain due to *stretching* forces, and the second term quantifies *compression* of a sensor between a module and its neighbor, if applicable. The first sum in the first term is over the counterclockwise (CCW) half of the module edges starting from node j, and the second term sums over the clockwise (CW) half of the module; k_m is the membrane spring constant (uniform for all modules and all states); ℓ^α and $\ell^{\alpha,rest}$ are the actual and rest lengths of the edges/springs along the membrane, and $\hat{\mathbf{u}}_j^\alpha$ and $\hat{\mathbf{v}}_j^\alpha$ are the vectors along the edges incident to vertex j. We include a weighting function, $\sigma(e)$, inspired by empirical evidence that deformations to the membrane do not transmit strain everywhere equally in the membrane; we assume that "influence" from a deformation drops off linearly as distance to the sensor increases, and integrate this linear weight over each edge to compute σ.

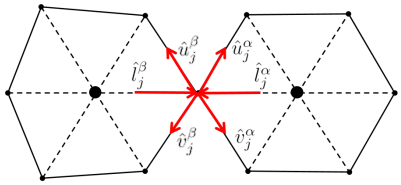


Fig. 3: Sketch of how local strain is computed as the sum of extension and compression forces acting on the sensor. The red arrows represent the force vectors in Eq. 1.

In the second term, k_i^β is the internal spring constant of robot β (a function of the target inflation level), $\ell_j^{\beta,rest}$ is the rest length of the internal spring between vertex j and the center of robot β, ℓ_j^β is the actual length of the internal spring, and $\hat{\boldsymbol{\ell}}_j^\beta$ is the unit vector from the center of robot β to vertex j. A is a system parameter denoting the relative weight of compression forces vs extension forces in sensor output, tuned through the validation described in Sec. IV-C, or is zero if no neighbor is present. To compute the simulated strain sensor output, we compute the magnitude of each term and sum the magnitudes, taking these as the normal and tangent forces to the sensor.

B. Simulating Actuation

Actuation is simulated by changing the rest lengths of the inner springs in the range (R_{min}, R_{max}), as well as the spring constants to reflect the calibration data in Section III. This method assumes that the radius of the robot increases linearly with time of inflation, so robot set points can be made in terms of target radius. We only operate our simulator in regimes where the actual length of the springs never goes below R_{min} (the radius when the robot is deflated), and thus we do not explicitly model the rigid centers of the robots.

Given a set of commands (radius setpoints per robot), the simulator discretizes time and solves the simulation at each timestep during the actuation, while logging sensor data.

This model easily admits some extensions to increase realism, such as increasing the discretization resolution with more springs per module, allowing agents to detach and attach when inter-module forces or proximity cross a threshold, or including dynamic effects for hardware with faster actuation or materials that admit more dynamics.

C. Simulator Validation

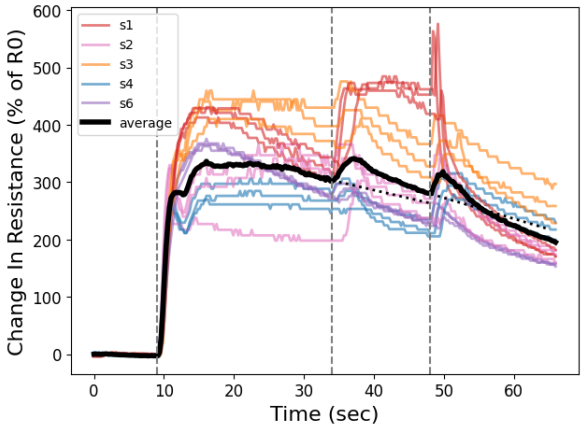
When validating the simulator and tuning parameters, our main goal was to replicate the average trends of the loaded strain sensors, as well as match the individual sensor responses as much as possible. This strategy permits us to test algorithms that use aggregate signals across all sensors on a single module. Real resistive sensors may increase in value both due to elongation (as the membrane expands into free space) and cross-sectional compression (when two modules squeeze against each other). Here, we confine our study to seven modules arranged in a compliant perimeter (Fig. 1), with a nominal inflation level of $0.5R_{max}$. Confining the collective helps guarantee a fixed contact topology of the robot collective, and simulates a larger collective since a module inflating on the boundary will experience resistance instead of simply inflating into free space without affecting the center module. We found that in the seven-robot configuration compression forces dominate, i.e. if a module inflates, the adjacent sensor on the neighboring module will increase in resistance; if it deflates, the resistance will decrease.

To validate the simulator, we performed a pre-programmed sequence in the seven-robot configuration with all modules initialized with 3.5s of inflation. Sensor data from the center ("listener") module was collected, while a single boundary ("active") module inflated to R_{max}, paused, then fully deflated (Fig. 4(a)). In Fig. 4(b) we compare real sensor signals with simulated sensor signals. For ease of visual comparison, the simulated sensor signals were scaled to match the real data's average response magnitude before the active module inflates. Note that sensor 5 was not recorded due to experimental setup constraints.

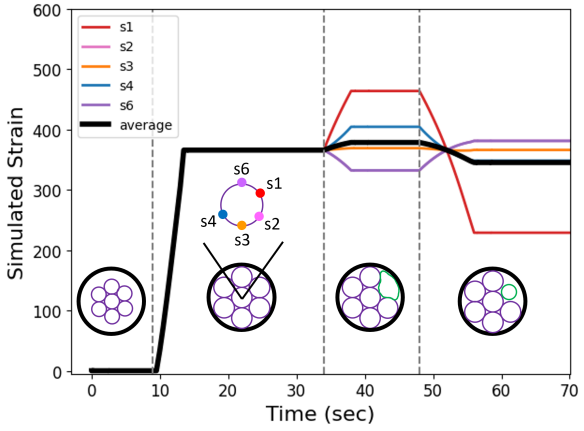
We qualitatively evaluated three sensor models: 1) a linear weighting term (σ in Eq.1) 2) an unweighted sum, or 3) including only the local forces from the two immediately adjacent springs. The linear weighting term gave the best match to observed data (average goes up during neighbor inflation, and down after, and sensor one increases in resistance while others stay flat or decrease). Using this model, we then tuned the relative magnitude of the compression and extension forces on the sensors by performing a parameter sweep over parameter A in Eq. 1, and compared the resulting simulated sensor data to hardware data, after which we chose to use $A = 1.3$ for the demonstrations in this paper.

Quantitatively, we compare the events in the real and simulated signals by extending a linear fit from the average signal in the two seconds before the inflation/deflation events (at 34 and 48 seconds). The resulting fits are extrapolated over the window of time following each event, as seen by the black dotted lines in Fig. 4(a). When the active agent inflates, the average signal over the time window $[34, 48]$ increases above this fit for the entire next 14 seconds, by an average of 18% of R_0 in the real data, and 12.7 units in simulation. When the active agent deflates, the average signal from hardware increases for six seconds before decreasing below the projected pre-event signal by an average of -8%, while in simulation the average signal decreases by 33 units. Clearly, more sophisticated system identification may be performed, but our goal for this work was to match the average behavior "close enough" to test the effectiveness of a proprioceptive algorithm.

There are three significant effects not captured in our simulator: 1) a slow decrease in resistance, likely caused by slow deflation and/or polymer relaxation effects; 2) transient effects related to perturbations causing sensor resistance to increase briefly during any type of "event"; and 3) each sensor has a unique offset due to the manufacturing process, while all simulated sensors are identical. To help compare simulated and real data with respect to the last point, we always normalize resistance data based on the average value of the sensor before the module inflates.



(a) Sensor data from three trials of the validation experiment. The first five seconds of this dataset were averaged to compute R_0 for each sensor, then the percent change was computed as $100 * R(t) - R_0)/R_0$. No other filtering or alignment was performed.



(b)

Fig. 4: Comparison of (a) hardware and (b) simulated sensors from the center listening module during the validation procedure, shown in insets. The red signals represent the sensor connected to the active agent. In each figure, the thick black line is the unweighted average of all the sensor responses.

V. CONSENSUS ALGORITHM

In [18], the authors present a generalized distributed consensus algorithm for modular robotic systems. They emphasise that in many cases, direct observation of the neighboring state is not possible (as is true in our system). They demonstrate sensor feedback laws that rely only on measurements of relative tilt or force between robotic modules, and give proofs of convergence in these cases. In this section, we extend this approach by approximating individual module strain and using differences between this quantity and a reference strain as a proxy for the information required for consensus algorithms.

A. Consensus

Consensus algorithms take the general form $x_i(t+1) = x_i(t) + \alpha \sum_{a_j \in N_i} g(\varepsilon_i, \varepsilon_j)$, where $x_i(t)$ is agent a_i's state at time step t, α is a small constant sometimes called the damping factor, N_i is the set of agent a_i's neighbors, ε_i is the sensor reading of agent a_i, and $g(\varepsilon_i, \varepsilon_j)$ is a sensory feedback function based on the interaction between agents a_i and a_j.

In our system, let the state $x_i(t)$ be the total strain felt by each Foambot, and we wish to design our system such that the collective maintains *strain consensus*, distributing environmental forces throughout the collective. Inflation generally increases local strain (and the apparent resistance of the robot's sensors), and deflation generally decreases strain. If the agents were able to directly communicate their measured strain value, we would define g as

$$g(\varepsilon_i, \varepsilon_j) = \varepsilon_j - \varepsilon_i \tag{2}$$

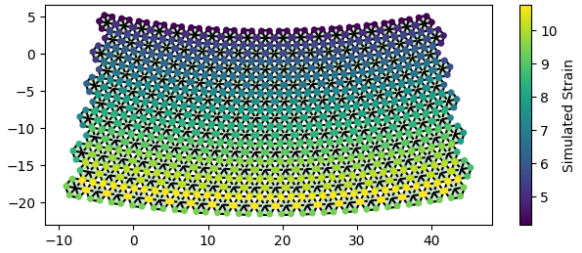
and the state update would be proportional to the *sum of differences* between a module's felt strain and all its neighbors' felt strain. However, if communication is not possible, we define g to be the sensor feedback law

$$g(\varepsilon_i, \varepsilon_j) = \varepsilon^* - \varepsilon_i \tag{3}$$

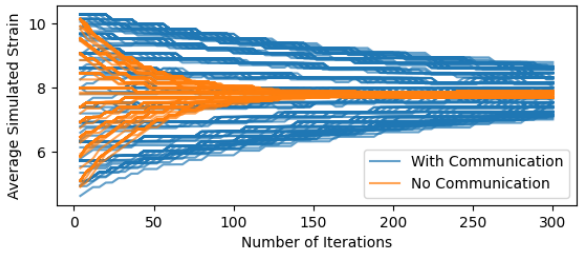
where ε_i is the average strain felt by agent i from all of its sensors and ε^* is a constant, pre-programmed reference strain. Note that Eq. 3 is a proportional control law, and no longer appears to be a true consensus algorithm. However, in dense collectives ε_i is implicitly a function of the neighboring modules' states via the inter-module interactions. As long as the strain signal ε_i is affected by the local strain, if g decreases in magnitude, the local strain is getting closer to ε^*. If g is negative, the local strain is larger than the reference strain, and the resulting feedback action will cause the robot to deflate (and vice versa if g is positive). Following the convergence proofs from [18], as long as ε^* admits a valid configuration, we should still expect all modules to eventually reach a consensus value, either the reference strain, or the nearest feasible strain value. Of course, this method is no longer guaranteed to reach the same equilibrium state as under traditional consensus, but it is much easier to implement, especially as we scale hardware down.

B. Algorithm Comparison

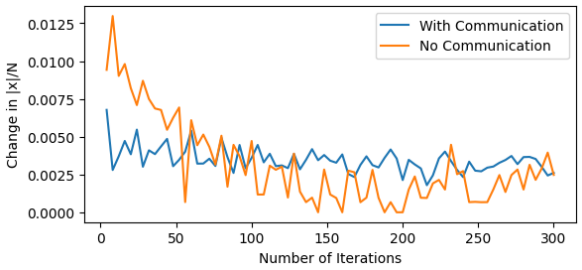
We test both feedback laws in our simulator using a large collective of 209 modules arranged in a rectangular sheet with 11 rows. The collective is initialized in a "bent" shape, created by inflating row i to $i/11$ of the maximum module radius. Fig. 5 shows a comparison of the behavior. For the proprioceptive controller, we used a reference strain of $\varepsilon^* = 7.8$, an approximation of the final state of the full-communication simulation. We set the damping factor $\alpha = 0.001$ for both. We include visualizations of the collective behavior under our controllers in the supplemental video; both approaches successfully "straighten" the sheet. Notably, the proprioceptive control approach acts faster, since the modules "know" the reference strain, while the communication-based consensus has a more constant rate of change per module. The non-monotonic behavior of the overall state change as



(a) The initial configuration of the 209-module assembly. Colorbar shows simulated sensor responses, nondimensional.



(b) The average strain per robot during iteration of both algorithms. The distinct groups of trajectories correspond to the different horizontal bands in (a).



(c) The change in average strain, computed every 4 iterations and averaged over all robots.

Fig. 5: Comparison of the consensus control laws, one that allows communication with neighbors (Eq. 2), and one that uses a proportional controller (Eq. 3) on proprioceptive data. Both collectives are initialized as seen in (a).

consensus progresses (Fig.5c) shows that the modules are being influenced by their neighbors, and the proportional controller does not totally dominate the observed dynamics. Further analysis of signal propagation dynamics, the effect of α, and convergence criteria are outside the scope of this work.

VI. CONCLUSIONS

We presented and characterized a new iteration of a soft, pneumatic, modular robot. Our main contributions are a low-dimensional, efficient model for estimating forces and strain in the system, with calibration and validation against our hardware system, as well as a demonstration of strain-mediated consensus using communication only through the substrate. In this case, the substrate is the robots' own bodies, which has exciting implications for smart materials that can exhibit collective behaviors without requiring centralized communication or even direct wireless communication between modules. It is also immediately apparent that the next step in this line of work is to apply more sophisticated proprioceptive algorithms to systems of coupled soft modules, such as onboard state-estimation over the history of sensor readings and actions, adaptive gain tuning, sensor fusion with other proprioceptive sensors such as pressure sensors, and programming collective behaviors through sensing of environmental signals such as gradients or EM fields. There are also many open questions in terms of robustness to noise and transient effects of real-world strain sensors.

REFERENCES

[1] K. H. Petersen and R. F. Shepherd, "Fluid-driven intrinsically soft robots," in *Robotic Systems and Autonomous Platforms*. Elsevier, 2019, pp. 61–84.

[2] H. Hamann, *Swarm robotics: A formal approach*. Springer, 2018.

[3] C. G. Vasquez and A. C. Martin, "Force transmission in epithelial tissues," *Developmental Dynamics*, vol. 245, no. 3, pp. 361–371, 2016.

[4] S. Ceron, M. A. Kimmel, A. Nilles, and K. H. Petersen, "Soft robotic oscillators with strain-based coordination," *IEEE Robotics and Automation Letters*, 2021.

[5] M. A. McEvoy and N. Correll, "Materials that couple sensing, actuation, computation, and communication," *Science*, vol. 347, no. 6228, 2015.

[6] N. J. Wilson, S. Ceron, L. Horowitz, and K. Petersen, "Scalable and robust fabrication, operation, and control of compliant modular robots," *Frontiers in Robotics and AI*, vol. 7, p. 44, 2020.

[7] M. Malley, B. Haghighat, L. Houe, and R. Nagpal, "Eciton robotica: Design and algorithms for an adaptive self-assembling soft robot collective," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4565–4571.

[8] R. M. McKenzie, M. E. Sayed, M. P. Nemitz, B. W. Flynn, and A. A. Stokes, "Linbots: Soft modular robots utilizing voice coils," *Soft robotics*, vol. 6, no. 2, pp. 195–205, 2019.

[9] S. Li, R. Batra, D. Brown, H.-D. Chang, N. Ranganathan, C. Hoberman, D. Rus, and H. Lipson, "Particle robotics based on statistical mechanics of loosely coupled components," *Nature*, vol. 567, no. 7748, pp. 361–365, 2019.

[10] P. Chvykov, T. A. Berrueta, A. Vardhan, W. Savoie, A. Samland, T. D. Murphey, K. Wiesenfeld, D. I. Goldman, and J. L. England, "Low rattling: A predictive principle for self-organization in active collectives," *Science*, vol. 371, no. 6524, pp. 90–95, 2021.

[11] M. A. Karimi, V. Alizadehyazdi, B.-P. Busque, H. M. Jaeger, and M. Spenko, "A boundary-constrained swarm robot with granular jamming," in *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*. IEEE, 2020, pp. 291–296.

[12] J. Boudet, J. Lintuvuori, C. Lacouture, T. Barois, A. Deblais, K. Xie, S. Cassagnere, B. Tregon, D. Brückner, J. Baret, *et al.*, "From collections of independent, mindless robots to flexible, mobile, and directional superstructures," *Science Robotics*, vol. 6, no. 56, 2021.

[13] W. Savoie, T. A. Berrueta, Z. Jackson, A. Pervan, R. Warkentin, S. Li, T. D. Murphey, K. Wiesenfeld, and D. I. Goldman, "A robot made of robots: Emergent transport and control of a smarticle ensemble," *Science Robotics*, vol. 4, no. 34, 2019.

[14] C. Della Santina, C. Duriez, and D. Rus, "Model based control of soft robots: A survey of the state of the art and open challenges," *arXiv preprint arXiv:2110.01358*, 2021.

[15] S. S. Ge, T. H. Lee, and Z. Wang, "Model-free regulation of multi-link smart materials robots," *IEEE/ASME transactions on mechatronics*, vol. 6, no. 3, pp. 346–351, 2001.

[16] D. Ma, S. Ceron, G. Kaiser, and K. Petersen, "Simple low-cost fabrication of soft sensors for feature reconstruction," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4049 – 4054, 2020.

[17] A. G. Fletcher, M. Osterfield, R. E. Baker, and S. Y. Shvartsman, "Vertex models of epithelial morphogenesis," *Biophysical journal*, vol. 106, no. 11, pp. 2291–2304, 2014.

[18] C.-H. Yu and R. Nagpal, "Self-adapting modular robotics: A generalized distributed consensus framework," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 1881–1888.