scientific reports

Check for updates

OPEN

Physical reservoir computing with origami and its application to robotic crawling

Priyanka Bhovad✉ & Suyi Li

A new paradigm called physical reservoir computing has recently emerged, where the nonlinear dynamics of high-dimensional and fixed physical systems are harnessed as a computational resource to achieve complex tasks. Via extensive simulations based on a dynamic truss-frame model, this study shows that an origami structure can perform as a dynamic reservoir with sufficient computing power to emulate high-order nonlinear systems, generate stable limit cycles, and modulate outputs according to dynamic inputs. This study also uncovers the linkages between the origami reservoir's physical designs and its computing power, offering a guideline to optimize the computing performance. Comprehensive parametric studies show that selecting optimal feedback crease distribution and fine-tuning the underlying origami folding designs are the most effective approach to improve computing performance. Furthermore, this study shows how origami's physical reservoir computing power can apply to soft robotic control problems by a case study of earthworm-like peristaltic crawling without traditional controllers. These results can pave the way for origami-based robots with *embodied mechanical intelligence*.

The animal kingdom is an endless source of inspiration for soft robotics[1,2]. Researchers have constructed compliant robots that can mimic all kinds of animal motions, like octopus locomotion[3], elephant trunk grasping[4], insect flying[5], jellyfish and fish swimming[6–8], as well as snake and insects crawling[9–11]. These robots share many similarities with animals regarding their shape and motion kinematics; however, their underlying sensing, actuation, and control architectures could be fundamentally different. Our engineered soft robots typically rely on a centralized controller (aka. an "electronic brain") that takes up all computing work to process sensor information, generate control commands, and make decisions. This approach often struggles to achieve high actuation speed and control effectiveness as soft robots exhibit virtually infinite degrees of freedom and complicated dynamic characteristics. On the other hand, animals have highly interconnected networks of nerves and muscles that can share the workload with the brain[12–14]. The animal body's morphology is an integral part of its actuation, control, and ultimately its "brain's" decision-making process, leading to far superior efficiency than our engineered soft robots.

Motivated by this disparity, an increasing number of researchers have embraced soft bodies' nonlinear dynamics as a computational resource to create an embodied intelligence and control[15–21]. As a result, a new computational paradigm called *morphological computation* has emerged in which the physical body of the robot itself takes part in performing low-level control tasks, such as locomotion coordination and modulation, to simplify the overall control architecture significantly[15–18,22]. The contributions of body morphology to cognition and control involve three major categories[20]: (1) Morphology facilitating control: wherein the physical design enables certain behaviors such as motion sequencing (e.g., passive dynamic walker[23]). (2) Morphology facilitating perception: wherein the physical design enables sensing (e.g., the nonuniform distribution of cells in the compound eyes of fly[24]). (3) Morphological computation, such as the *physical reservoir computing* (PRC), wherein a physical body performs genuine computations. Among these, physical reservoir computing shows promising potentials because of its balanced simplicity and versatility to perform applicable computation with encoding and decoding[20].

Reservoir computing is a computational framework based on artificial recurrent neural networks (RNNs), which have been used extensively for problems involving time-series prediction like the stock market and weather forecasting, robotic motion planning and control, text and speech recognition[21,25–31]. In RNNs, the output of the current time step depends on the results from the previous time step in addition to the current input. Since RNNs involve both forward and back-propagation of input data, training them became a challenging task. To address this difficulty, Jaeger introduced the concept of a *fixed* recurrent neural network as Echo State Networks (ESNs)[25], and Maass introduced Liquid State Machines (LSMs)[26]. Later, these two concepts merged under the

Department of Mechanical Engineering, Clemson University, Clemson, SC, USA. ✉email: pbhovad@clemson.edu

umbrella of reservoir computing (RC). In RC, the neural network (aka. the "reservoir") has fixed interconnections and input weights, and only the linear output readout weights are trained by simple techniques such as, linear or ridge regression. The reservoir's dynamics transforms the input data stream into a high-dimensional state space, capturing its nonlinearities and time-dependent information for computation tasks.

More importantly, the reservoir's fixed nature opens up the possibility of using physical bodies—such as a random network of nonlinear spring and mass oscillators[18,32,33], tensegrity structures[15–17,34], and soft robotic arms[19,35,36]—to conduct computation, hence the paradigm of *Physical* Reservoir Computing. These physical systems have been shown to possess sufficient computational power to achieve complex computing tasks e.g. emulating other nonlinear dynamic systems, pattern generation[17–19,21,32,34], speech recognition[37], and machine learning[21,31,33,36]. More importantly, robotic bodies with sufficient nonlinear dynamics can also perform like a physical reservoir and directly generate locomotion gait without using the traditional controllers[17,21,38–40]. Despite the recent progress, physical reservoir computing is still a nascent field, and it is worthwhile to examine the computing power of a wide variety of different compliant mechanical systems, especially those with broad application potential in soft robotic locomotion and intelligent structures. These dynamic physical reservoirs with embedded feedback can function as soft robotic skeletons, and simultaneously generate and maintain the periodic trajectories essential for animal-inspired locomotion gait generation.

One such compliant mechanism that has garnered much attention over recent years is Origami – a traditional play of folding paper into sophisticated and three-dimensional shapes. Origami mechanisms are compact, easy to fabricate, and scale-independent (aka. Origami robots can be fabricated at different scales but still follow similar folding principles[41–44]). Origami mechanisms with complex crease folding patterns and integrated actuator-sensor network exhibit many desirable soft body properties. Similar to the soft and compliant biological systems, origami features non-linear kinematics and dynamics, non-linearly varying stiffness, large deformation range, compliance, and shape-morphing capability. The origami mechanisms are stiff enough to be used as structural skeleton for robots, and at the same time are compliant enough to provide large deformation range required for soft robotic locomotion. Over the past decades, origami has evolved into an engineering framework for constructing multi-transformable deployable structures[44–47], advanced meta-materials and shape morphing structures[41,43,48–53]. It is already a popular engineering platform for constructing soft robotic skeletons that mimic wide range of animal motions, e.g. worm-like crawling, insect-like walking, and octopus arm-inspired manipulation[9,42,54–63]. Moreover, the nonlinear mechanics and dynamics induced by folding could also enhance robotic performance[64–66]. Thus, we investigate the use of origami as a physical reservoir and show that origami based robotic skeleton can indeed generate the periodic patterns for autonomous locomotion gait generation.

We show that origami's nonlinear folding dynamics possesses significant computing power, which could add a valuable dimension to the field of origami-based engineering. A mechanical system must exhibit several essential properties to perform as a reservoir[21]. The first one is high-dimensionality, which allows the reservoir to gather as much information possible from the input data stream, separating its spatio-temporal dependencies and projecting it onto a high-dimensional state-space. The second one is nonlinearity so that the reservoir acts as a nonlinear filter to map the information from the input stream. All the computation complexity is associated with this nonlinear mapping, thus training the linear static readout becomes a straightforward task. The third one is fading memory (or short-term memory), ensuring that only the recent input history influences the current output. The fourth one is separation property to classify and segregate different response signals correctly, even with small disturbances or fluctuations. Moreover, if two input time series differed in the past, the reservoir should produce different states at subsequent time points[67]. Our physics-informed numerical simulations prove that origami inherently satisfies these four requirements and can complete computation tasks like emulation, pattern generation, and output modulation.

Moreover, we conduct extensive numerical simulations to uncover the linkage between origami design and its computing power, providing the guideline to optimize computing performance. Finally, we demonstrate how to directly embed reservoir computing in an origami robotic body to generate earthworm-like peristalsis crawling without using any traditional controllers. This study's results could foster a new family of origami-based soft robots that operate with simple mechatronics, interact with the environment through distributed sensor and actuator networks, and respond to external disturbances by modulating their activities.

Constructing the origami reservoir

In this study, we construct a physical reservoir using the classical Miura-ori sheets. We can easily modify basic Miura-ori geometry to create complex structures such as curved Miura-ori surfaces, stacked Miura-ori, or origami-tubes with various cross sections. We show that origami can indeed act as a reservoir and even simplest of origami pattern can be turned into peristaltic crawling robot powered by reservoir computing. Miura-ori is essentially a periodic tessellation of unit cells, each consisting of four identical quadrilateral *facets* with *crease* lengths a, b and an internal sector angle γ (Fig. 1a)[48,68]. The folded geometry of Miura-ori can be fully defined with a dihedral *folding angle* θ ($\in [0, \pi/2]$) between the x-y reference plane and its facets (Fig. 1b). The reservoir size is defined as $n \times m$, where n and m are the number of origami *nodes* (aka. vertices where crease lines meet) in x and y-directions, respectively. N is the total number of creases in the origami reservoir.

Dynamics modeling of the origami. To investigate this origami reservoir's computing capacity, one must first obtain its time responses under dynamic excitation. To this end, we adopt and expand the lattice framework approach to simulate its nonlinear dynamics[68–70]. In this approach, origami creases are represented by pin-jointed stretchable truss elements with prescribed spring coefficient K_s. Folding (or bending) along the crease line is simulated by assigning torsional spring coefficient K_b. We further triangulate the quadrilateral facets with additional 'virtual' truss elements to estimate the facet bending with additional torsional stiffness (typi-

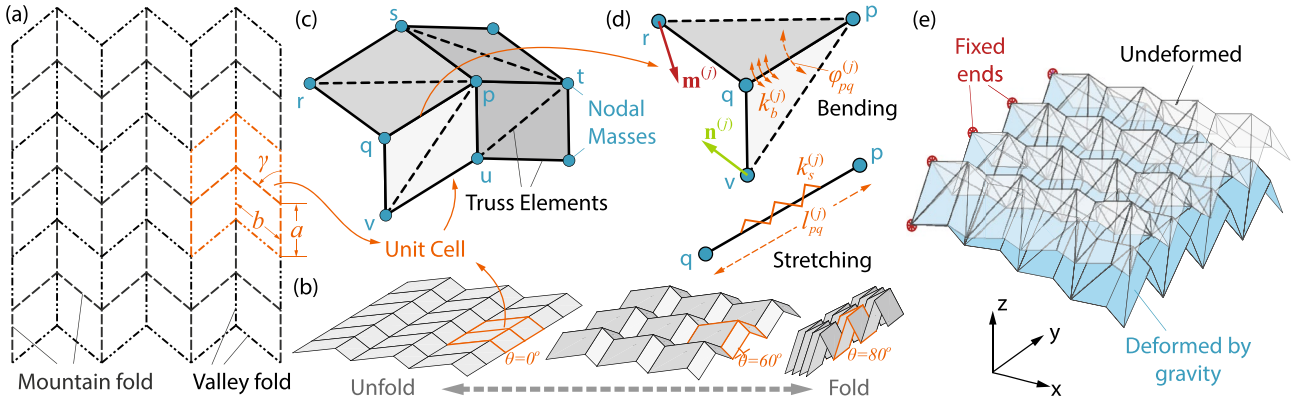

Figure 1. The nonlinear Truss-frame approach for simulating the origami dynamics. (**a**) The crease pattern of the classical Miura-ori, with a unit cell highlighted. (**b**) The rigid-folding kinematics of the Miura-ori. (**c**) The truss-frame approach discretizes the Miura-ori unit cell, showing the distribution of truss elements along the creases and across the facets, as well as the nodal masses. (**d**) Detailed kinematics and mechanics set up to analyze the bending and stretching along the truss #*pq*. Notice that $\mathbf{m}^{(j)}$ and $\mathbf{n}^{(j)}$ are the current surface normal vectors defined by triangles #*pqr* and #*pqv*, respectively. (**e**) The bending of the Miura-ori sheet under its weight. This simulation serves to validate appropriate material property assignments.

cally, K_b across the facets is larger than those along the creases). The rationale behind adding these 'virtual' facet bending creases can be explained with the example of Miura-ori structure. The ideal Miura-ori structure is rigid foldable when the in-plane folding motion is the primary deformation mode. Meanwhile, Miura-ori also exhibits out-of-plane twisting and saddle-shaped deformations that can be prominent in dynamic responses[48,68]. More importantly, these non-rigid folding deformations are desirable for reservoir computing. The presence of two out-of-plane deformation modes depends mainly on the ratio between facet bending stiffness ($k_{b,f}$) and crease folding stiffness (k_b). When $k_{b,f}/k_b \gg 1$, we can assume Miura-ori to be rigid-foldable. When $k_{b,f}$ and k_b are comparable, out-of-plane deformation modes can be significant. The non-rigid foldable behavior of many origami mechanisms and metastructures has been studied with this approach, for example, square twist pattern[71], Kresling pattern[64], and Miura-ori pop-through defect[69].

In essence, lattice framework approach discretizes the continuous origami sheet into a network of pin-jointed truss elements connected at the nodes (Fig. 1c). A typical reservoir consists of an interconnected network of units governed by nonlinear dynamics. The origami reservoir, in this case, consists of a network of nodes with their interconnections defined by the underlying crease pattern. It's important to note that our approach does not include the nonlinear/ hyperelastic material constitutive model, the effects due to finite material thickness, viscosity, and nonlinear stiffness changes, etc. The corresponding governing equations of motion, in terms of node #p's displacement ($\mathbf{x}_p$) as an example, are:

$$m_p \ddot{\mathbf{x}}_p^{(j)} = \mathbf{F}_p^{(j)}, \tag{1}$$

where the superscript "(j)" represents the jth time step in numerical simulation, and m_p is the equivalent nodal mass. Unless noted otherwise, the mass of the origami sheet is assumed to be equally distributed to all its nodes. $\mathbf{F}_p^{(j)}$ is the summation of internal and external forces acting on this node in that

$$\mathbf{F}_p^{(j)} = \sum \mathbf{F}_{s,p}^{(j)} + \sum \mathbf{F}_{b,p}^{(j)} + \mathbf{F}_{d,p}^{j} + \mathbf{F}_{a,p}^{(j)} + m_p \mathbf{g}, \tag{2}$$

where the five terms on the right hand side are the forces from truss stretching, crease/facet bending, equivalent damping, external actuation, and gravity, respectively. The formulation of these forces are detailed below.

Truss stretching forces. The truss elements are essentially elastic springs with axial stretching stiffness ($K_s^{(j)} = EA/l^{(j)}$). Here, EA is the material constant, and $l^{(j)}$ is the truss element's length at the current jth time step. Thus, the axial stiffness is updated at each time-step, accommodating the truss element's increase in stiffness as it is compressed and vice-a-versa. The stretching force from a truss connecting node #p and one of its neighbouring nodes #q is,

$$\mathbf{F}_{s,p}^{(j)} = -K_s^{(j)} \left(l_{pq}^{(j)} - l_{pq}^{(0)} \right) \frac{\mathbf{r}_p^{(j)} - \mathbf{r}_q^{(j)}}{|\mathbf{r}_p^{(j)} - \mathbf{r}_q^{(j)}|} \tag{3}$$

where $l_{pq}^{(0)}$ is the truss length at its initial resting state. $\mathbf{r}_p^{(j)}$ and $\mathbf{r}_q^{(j)}$ are the current position vectors of these two nodes, respectively. To calculate the total truss stretching forces acting on node #p, similar equations apply to all of its neighbour nodes through trusses (e.g., node q, r, s, t, u, and v in Fig. 1c).

Crease/facet bending forces. The crease folding and facet bending are simulated with torsional spring coefficient ($K_b^{(j)} = k_b l^{(j)}$), where k_b is torsional stiffness *per unit length*. Here, we adopt the formulation developed by Liu and Paulino[69]. For example, if the stored potential energy due to crease folding along the truss between #p and #q is: $\mathbf{u}_{pq}^{(j)} = \frac{1}{2}K_b^{(j)}(\varphi_{pq}^{(j)} - \varphi_{pq}^{(0)})^2$, then the force acting on nodes #p is:

$$\mathbf{F}_{b,p}^{(j)} = -\frac{\partial \mathbf{u}_{pq}^{(j)}}{\partial \varphi_{pq}^{(j)}}\frac{\partial \varphi_{pq}^{(j)}}{\partial \mathbf{r}_p^{(j)}} = -K_b^{(j)}(\varphi_{pq}^{(j)} - \varphi_{pq}^{(0)})\frac{\partial \varphi_{pq}^{(j)}}{\partial \mathbf{r}_p^{(j)}} \tag{4}$$

where $\varphi_{pq}^{(j)}$ is the current dihedral angle along truss pq (aka. the dihedral angle between the triangles #pqr and #pqv in Fig. 1d), and $\varphi_{pq}^{(0)}$ is the corresponding initial value. $\varphi_{pq}^{(j)}$ can be calculated as

$$\varphi_{pq}^{(j)} = \eta \arccos\left(\frac{\mathbf{m}^{(j)} \cdot \mathbf{n}^{(j)}}{|\mathbf{m}^{(j)}||\mathbf{n}^{(j)}|}\right) \text{ modulo } 2\pi \tag{5}$$

$$\eta = \begin{cases} \text{sign}\left(\mathbf{m}^{(j)} \cdot \mathbf{r}_{pv}^{(j)}\right), & \mathbf{m}^{(j)} \cdot \mathbf{r}_{pv}^{(j)} \neq 0 \\ 1. & \mathbf{m}^{(j)} \cdot \mathbf{r}_{pv}^{(j)} = 0 \end{cases} \tag{6}$$

Here, $\mathbf{m}^{(j)}$ and $\mathbf{n}^{(j)}$ are current surface normal vector of the triangles #pqr and #pqv, respectively, in that $\mathbf{m}^{(j)} = \mathbf{r}_{rq}^{(j)} \times \mathbf{r}_{pq}^{(j)}$ and $\mathbf{n}^{(j)} = \mathbf{r}_{pq}^{(j)} \times \mathbf{r}_{pv}^{(j)}$. In addition, $\mathbf{r}_{pq}^{(j)} = \mathbf{r}_p^{(j)} - \mathbf{r}_q^{(j)}$, $\mathbf{r}_{rq}^{(j)} = \mathbf{r}_r^{(j)} - \mathbf{r}_q^{(j)}$, and $\mathbf{r}_{pv}^{(j)} = \mathbf{r}_p^{(j)} - \mathbf{r}_v^{(j)}$. This definition of $\varphi_{pq}^{(j)}$ ensures that the folding angle for valley crease lies in $(0, \pi]$ and the folding angle for mountain crease lies in $(\pi, 2\pi]$. The derivative between folding angle $\varphi_{pq}^{(j)}$ and the nodal #p's current position vector is

$$\frac{\partial \varphi_{pq}^{(j)}}{\partial \mathbf{r}_p^{(j)}} = \left(\frac{\mathbf{r}_{pv}^{(j)} \cdot \mathbf{r}_{pq}^{(j)}}{|\mathbf{r}_{pq}^{(j)}|^2} - 1\right)\frac{\partial \varphi_{pq}^{(j)}}{\partial \mathbf{r}_v^{(j)}} - \left(\frac{\mathbf{r}_{rq}^{(j)} \cdot \mathbf{r}_{pq}^{(j)}}{|\mathbf{r}_{pq}^{(j)}|^2}\right)\frac{\partial \varphi_{pq}^{(j)}}{\partial \mathbf{r}_r^{(j)}} \tag{7}$$

where

$$\frac{\partial \varphi_{pq}^{(j)}}{\partial \mathbf{r}_r^{(j)}} = \frac{|\mathbf{r}_{pq}^{(j)}|}{|\mathbf{m}^{(j)}|^2}\mathbf{m}^{(j)}, \tag{8}$$

$$\frac{\partial \varphi_{pq}^{(j)}}{\partial \mathbf{r}_v^{(j)}} = -\frac{|\mathbf{r}_{pq}^{(j)}|}{|\mathbf{n}^{(j)}|^2}\mathbf{n}^{(j)}. \tag{9}$$

Again, to calculate the total crease folding and facet bending forces acting on node #q, similar equations apply to trusses connected to this node (e.g., truss pq, pr, ps, pt, pu, and pv in Fig. 1d).

Damping forces. Estimating damping ratio and damping force is essential to achieve realistic dynamic responses and reduce numerical simulation error accumulation. In this study, we follow the formulation developed in[70,72]. This formulation first calculates the average velocity of a node with respect to its neighboring nodes ($\mathbf{v}_{\text{avg}}^{(j)}$) to effectively remove the rigid body motion components from the relative velocities and ensure that these components are not damped. Then damping force $\mathbf{F}_{d,p}^{(j)}$ applied on node #p is given by

$$\mathbf{F}_{d,p}^{(j)} = -c_d^{(j)}(\mathbf{v}_p^{(j)} - \mathbf{v}_{\text{avg}}^{(j)}) \tag{10}$$

$$c_d^{(j)} = 2\zeta\sqrt{K_s^{(j)} m_p} \tag{11}$$

where $c_d^{(j)}$ is the equivalent damping coefficient, and ζ is the damping ratio.

Actuation force. In the origami reservoir, two types of creases receive actuation. The first type is "input creases," and they receive input signal $u(t)$ required for emulation and output modulation tasks. The second type is "feedback creases," and they receive reference or current output signal $z(t)$ required by all computing tasks in this study except for the emulation task. In the case of multiple outputs, different groups of feedback creases are present. Here, the selection of input and feedback creases are random. There are many methods to implement actuation to deliver input $u(t)$ and reference/feedback signal $z(t)$ to the reservoir. For example, the actuation can take the form of nodal forces on a mass-spring-damper network[18,32], motor generated base rotation on octopus-inspired soft arm[19], or spring resting length changes in a tensegrity structure[34]. In origami, the actuation can take the form of moments that can fold or unfold the selected creases. We assume that the resting angle $\varphi^{(0)}$ of the input and feedback creases will change—in response to the actuation at every time step—to a new equilibrium $\varphi_{a,0}^{(j)}$ in that[34,73]

$$\varphi_{a,0}^{(j)} = W_{\text{in}} \tanh(u^{(j)}) + \varphi^{(0)} \quad \text{for input creases;} \tag{12}$$

$$\varphi_{a,0}^{(j)} = W_{\text{fb}} \tanh(z^{(j)}) + \varphi^{(0)} \quad \text{for feedback creases.} \tag{13}$$

where W_{in} and W_{fb} are the input and feedback weight associated with these actuated creases. They are assigned before the training and remain unchanged after that. $u^{(j)}$ and $z^{(j)}$ are the input and feedback signal at the jth time step. The magnitude of W_{in} and W_{fb} are selected such that $\varphi_{a,0}^{(j)} \in [0, 2\pi)$ and consistent with the folding angle assignment. This approach of assigning new equilibrium folding angles is similar to traditional neural network studies that use tanh as a nonlinear activation function to transform function $z(t)$ into a new one with magnitudes between $[-1, 1]$. Additionally, it prevents actuator saturation due to spurious extreme values of $z(t)$. Denote the torsional stiffness of actuated creases by $K_{b,a}^{(j)}$, and we can update Equation (4) for the actuated creases (using node #p as an example)

$$\mathbf{F}_{a,p}^{(j)} = -K_{b,a}^{(j)} \left(\varphi_{pq}^{(j)} - \varphi_{a,0,pq}^{(j)} \right) \frac{\partial \varphi_{pq}^{(j)}}{\partial \mathbf{r}_p^{(j)}}, \tag{14}$$

The calculation of other terms in this equation are the same as those in the force from crease folding and facet bending. In this work, we focus on compliant sheet materials that can easily fold into origami shapes and provide sufficient compliance for soft robotic motions. We also want to ensure that our numerical simulation results are applicable to different material selections. The range of material parameters in aforementioned equations is set according to such considerations. For example, the k_s and k_b values in the baseline designs come from (1) averaged results from our prior experiments using folded thick paper and PET polymer sheets[63,66], and (2) a careful survey of relevant literature[45,48,49,68,69,71,74]. It is worth noting that obtaining the equivalent k_s and k_b values is not trivial, and it depends on many material and geometric factors, including the origami facets' size and sheet material thickness, etc. Later in the parametric studies, we choose a relatively wide range of material properties to accommodate for such complexity. More importantly, we can also ensure the reservoir computing result can apply to origami reservoirs with different polymer/plastic-like material selections.

Once the governing equations of motion are formulated with above considerations, they are solved using MATLAB's `ode45` solver with 10^{-3} second time-steps. As an example, we show a simulation of the Miura-ori sheet deformation under its own weight (Fig. 1e). Although the governing equation of motions use nodal displacement $\mathbf{x}^{(j)}$ as the independent variables, we use the dihedral crease angles $\varphi^{(j)}$ as the *reservoir state* variables to characterize the origami's time responses. This is because measuring crease angles is easier to implement by embedded sensors, and $\varphi^{(j)}$ can be directly calculated from $\mathbf{x}^{(j)}$ via the Equations 5 and 6.

Setting up reservoir computing. Similar to the actuated creases (aka. input creases and feedback creases), we designate "sensor creases" for measuring the reservoir states. We denote N_a as the number of actuated creases, and N_s for sensor creases. It is worth noting that, the actuated creases are typically small subset of all origami creases (i.e., $N_a < N$). The sensor creases, on the other hand, can be all of the origami creases ($N_s = N$) or a small subset as well ($N_s < N$).

Once the selection of input, feedback, and sensor creases is completed, one can proceed to the computing. Physical reservoir computing for tasks that require feedback, and output modulation consists of two phases: The "training phase" and "closed-loop phase." While the emulation tasks require the training phase only.

Training phase. In this phase, we use the teacher forcing to obtain the readout weights $\mathbf{W}_{\text{out}}$ corresponding to every reservoir state (aka. the dihedral angles of the sensor creases). Suppose one wants to train the reservoir to generate a nonlinear time series $z(t)$ (aka. the reference output). The feedback creases receive the reference output and it dynamically excites the origami reservoir under an open-loop condition without feedback (Fig. 2a). The reservoir states $\varphi^{(j)}$ at every time step are measured and then compiled into a matrix $\mathbf{\Phi}$.

Once the numerical simulation is over, we segregate the reservoir state matrix $\mathbf{\Phi}$ into the washout step, training step, and testing step. The washout step data is discarded to eliminate the initial transient responses. We then calculate the output readout weights $\mathbf{W}_{\text{out}}$ using the training step data via simple linear regression:

$$\mathbf{W}_{\text{out}} = [\mathbf{1}\ \mathbf{\Phi}]^{+}\mathbf{Z} = \bar{\mathbf{\Phi}}^{+}\mathbf{Z} \tag{15}$$

where, $[.]^{+}$ refers to the Moore–Penrose pseudo-inverse to accommodate non-square matrix. $\mathbf{1}$ is a column of ones for calculating the bias term $W_{\text{out},0}$ to shift the fitted function when necessary. $\mathbf{Z}$ contains the reference signals at each time step, and it is a matrix if more than one references are present. Lastly, we use testing step data to verify reservoir performance. It is worth noting that white noise of amplitude 10^{-3} is superimposed on the reservoir state matrix during training to ensure the robustness of the readout result against numerical imperfections, external perturbations[32], and instrument noise in "real-world" applications.

Closed-loop phase. Once the training phase is over and readout weights are obtained, we run the reservoir in the closed-loop condition. That is, instead of using the reference output $z(t)$, the current output $z^{*}(t)$ is sent to the feedback creases (Fig. 2b), and

$$z^{*}(t) = W_{\text{out},0} + \sum_{i=1}^{N_s} W_{\text{out},i}\varphi_i(t) = \mathbf{W}_{\text{out}}^{T}\bar{\mathbf{\Phi}} \tag{16}$$

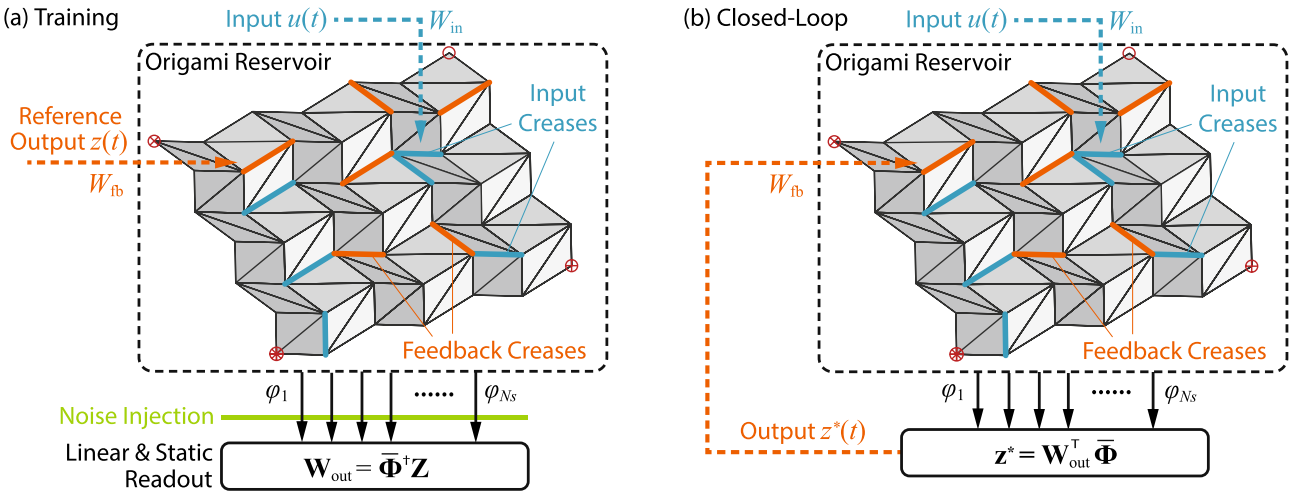

Figure 2. The setup of physical reservoir computing with origami. (**a**) The training phase. The feedback creases receive the reference (or targeted) output $z(t)$; while white noise is added to the reservoir state matrix $\bar{\mathbf{\Phi}}$ before calculating output weights $\mathbf{W}_{out}$; (**b**) The closed-loop phase. The output weights obtained in the training phase are used to calculate the current output, which is fed back to the feedback creases.

Parameter	Value
Reservoir size and material properties	
Size	9×9
Nodal mass	7 g
EA	100 N
$k_{b,a}$	1 N/(m-rad)
k_b	0.2525 N/(m-rad)
Facets k_b	10 N/(m-rad)
ζ	0.2
Geometric design of Miura-ori	
a	16 mm
b	10 mm
γ	50°
θ	60°
Actuator and sensor creases	
No. of sensors (N_s)	N
No. of actuators (N_a)	$0.45N$
No. of Feedback creases	$0.3N$
No. of Input creases	$0.15N$

Table 1. Design of a baseline origami reservoir in this study.

where, N_s is the number of sensor creases, and $\bar{\mathbf{\Phi}} = [\mathbf{1}\ \mathbf{\Phi}]$. Thus, the reservoir runs autonomously in the closed-loop phase without any external interventions.

We study the closed loop performance of reservoir by calculating the Mean Squared Error (MSE) using M time-steps as follows:

$$\text{MSE} = \frac{1}{M}\sum_{j=1}^{M}\left(z(j) - z^*(j)\right)^2 \tag{17}$$

To estimate performance when multiple reference outputs are present, we combine the MSEs by taking a norm over the individual MSEs.

Computation tasks by the origami reservoir

In this section, we use the origami reservoir to emulate multiple nonlinear filters simultaneously, perform pattern generation, and modulate outputs. The baseline variables for the origami geometric design, material properties, and reservoir parameters are given in Table 1.

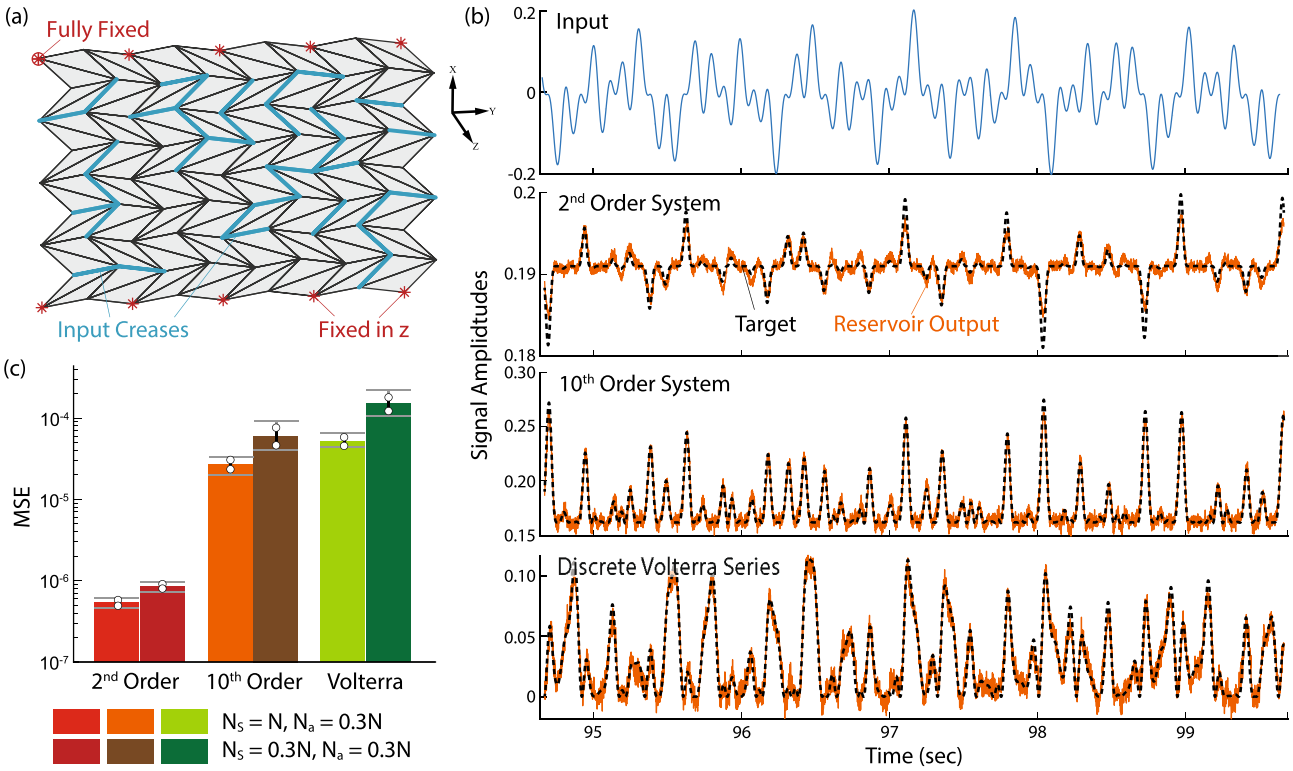

Figure 3. Emulation tasks with the origami reservoir. (**a**) The Miura-ori reservoir used for this task with input creases highlighted. Appropriate boundary conditions are also necessary. (**b**) Examples of trajectories generated in the emulation task including (from top to bottom) input signal $u(t)$, 2nd order, 10th order system, and Volterra series. Dashed curves are the targeted trajectories, and solid curves are the result of the reservoir. (**c**) Error analysis of the emulation tasks. Circles are the standard deviation of MSE, and horizontal bars are the corresponding extreme values.

Emulation task. This sub-section shows that the origami reservoir can emulate multiple nonlinear filters simultaneously using a single input. Such emulation is a benchmark task for evaluating the performance in RNN training[75] and prove the multi-tasking capability of physical reservoirs[18,19]. Note that the emulation task involves only the training phase, so there are no feedback creases in this case. Consequently, we excite the reservoir by sending the input function $u(t)$ to the input creases and train it to find three sets of readout weights in parallel via linear regression. Here, $u(t)$ is a product of three sinusoidal functions with different frequencies, and the three target nonlinear filters are 2nd-order nonlinear dynamic system $z_1(t)$, a 10th-order nonlinear dynamic system $z_2(t)$, and discrete Volterra series $z_3(n)$ (detailed in Table 2).

We use a 9×9 Miura-ori reservoir shown in Fig. 3a in this task, exciting the reservoir from complete rest and training it for 100 seconds. We discard the first 50 seconds of data as the washout step, use the data from the next 45 seconds to calculate the optimum static readout weights, and then use the last 5 seconds of data to calculate the MSE for performance assessments. Results in Fig. 3b show that the origami reservoir can emulate these three nonlinear filters. As the nonlinearity and complexity of the nonlinear filter increases, MSE also increases (Fig. 3c).

Moreover, we compare the emulation performance when all N creases are used as sensor creases versus when only actuated creases are used as sensors ($N_s = N_a = pN$). The increase in MSE is marginal in the latter case. Therefore, the origami satisfies the previously mentioned nonlinearity and fading memory requirements to be a physical reservoir, and one only needs to use the input creases angles as the reservoir states to simplify the reservoir setup.

We also experimentally demonstrate the emulation task using a paper-based origami reservoir (Fig. 4 and Supplemental Video B). Details of this experiment are summarized in "Methods and materials".

Pattern generation task. Pattern generation tasks are essential for achieving periodic activities such as robotic locomotion gait generation and manipulator control where persistent memory is required. That is, by embedding these patterns (or limit cycles) in the origami reservoir, one can generate periodic trajectories in the closed-loop. We again use a 9×9 Miura-ori reservoir and randomly select 30% of its creases as the feedback creases (Fig. 5a). This task does not require input creases. These feedback creases are divided into two groups for the two components of 2D trajectories. We run the training phase for 100 seconds for each pattern, discard the initial 15 seconds of data as the washout step and use the next 51 seconds' data to calculate the optimum output readout weights.

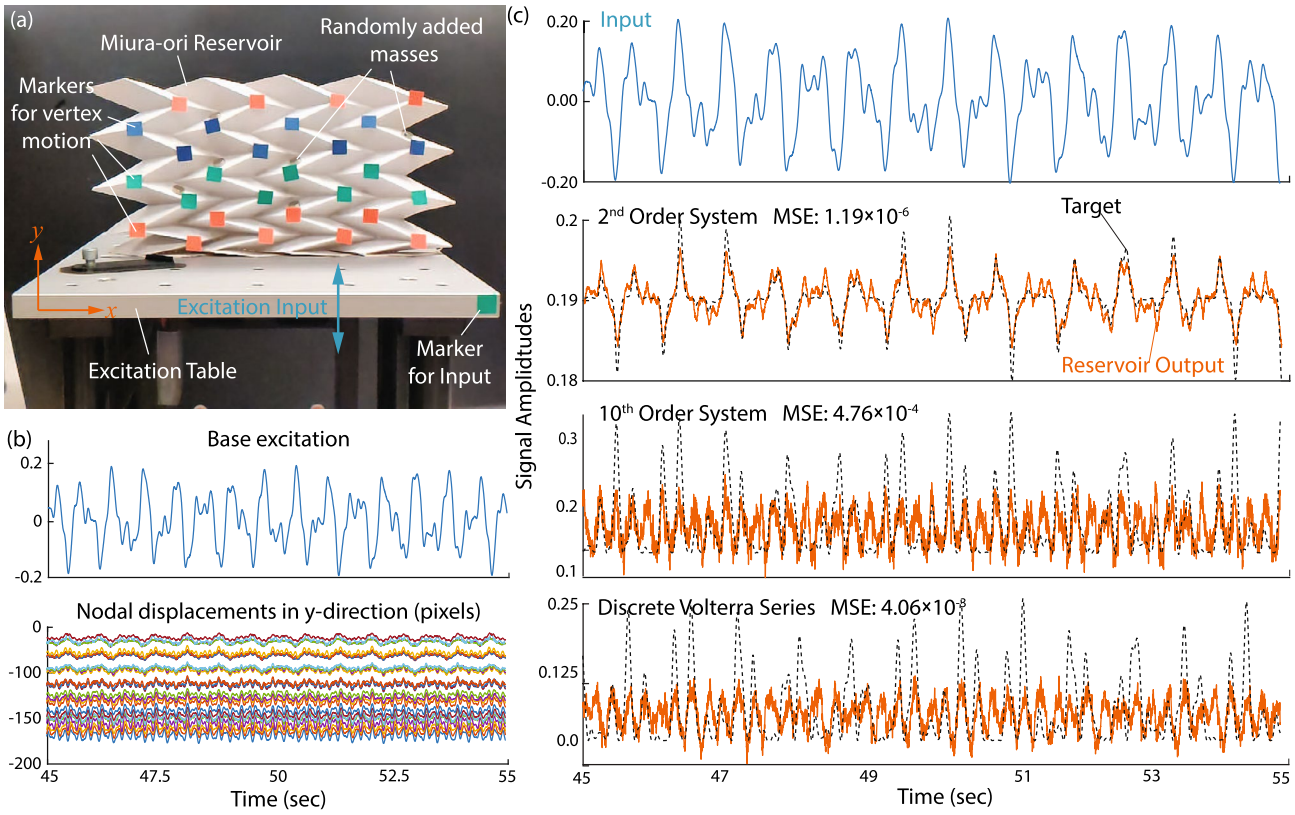

Figure 4. Proof-of-concept experiment showing the emulation task with the origami reservoir. (**a**) The experimental setup depicting the excitation table and paper-based Miura-ori reservoir with sensor node markers attached. The base of the reservoir is fixed to the base plate of the exciter. Notice that additional mass are added at random locations to break the origami's symmetry. (**b**) The base-plate displacement/input signal (u(t)) and recorded nodal displacements in y-direction. (**c**) Examples of trajectories generated in the emulation task including (from top to bottom) input signal u(t), 2nd-order, 10th-order system, and Volterra series. Dashed curves are the targeted trajectories, and solid curves are the result of the reservoir.

Type	Functions in discretized form (at *j*th time step)
Input	$u(j) = 0.2\sin(2\pi f_1 j\Delta t)\sin(2\pi f_2 j\Delta t)\sin(2\pi f_3 j\Delta t)$
	$f_1 = 2.11$ Hz, $f_2 = 3.73$ Hz, $f_3 = 4.33$ Hz, $\Delta t = 10^{-3}$
2nd order system	$z_1(j+1) = 0.4z_1(j) + 0.4z_1(j)z_1(j-1) + 0.6(u(j\Delta t))^3 + 0.1$
10th-order system	$z_2(j+1) = 0.3z_2(j-1) + 0.05z_2(j-1)\sum_{i=1}^{10} z_2(j-i)$
	$+1.5u((j-10)\Delta t)u((j-1)\Delta t) + 0.1$
Discrete Volterra series	$z_3(j+1) = 100\sum_{\tau_1=0}^{T}\sum_{\tau_2=0}^{T} h(\tau_1,\tau_2)u(j-\tau_1)u(nj-\tau_2)$
	$h(\tau_1,\tau_2) = \exp\left(\frac{(\tau_1\Delta t-\mu_1)^2}{2\sigma_1^2} + \frac{(\tau_2\Delta t-\mu_2)^2}{2\sigma_2^2}\right)$
	$\mu_1 = \mu_2 = 0.1, \sigma_1 = \sigma_2 = 0.05$

Table 2. Emulation task functions.

Generating nonlinear limit cycles. In the following results, the origami reservoir demonstrates its computation capability via generating quadratic limit cycle, Van der Pol oscillator, and the Lissajous curve in closed-loop. The quadratic limit cycle is defined by two differential equations:

$$\dot{x}_1 = x_1 + x_2 - \epsilon(t)x_1\left(x_1^2 + x_2^2\right), \tag{18}$$

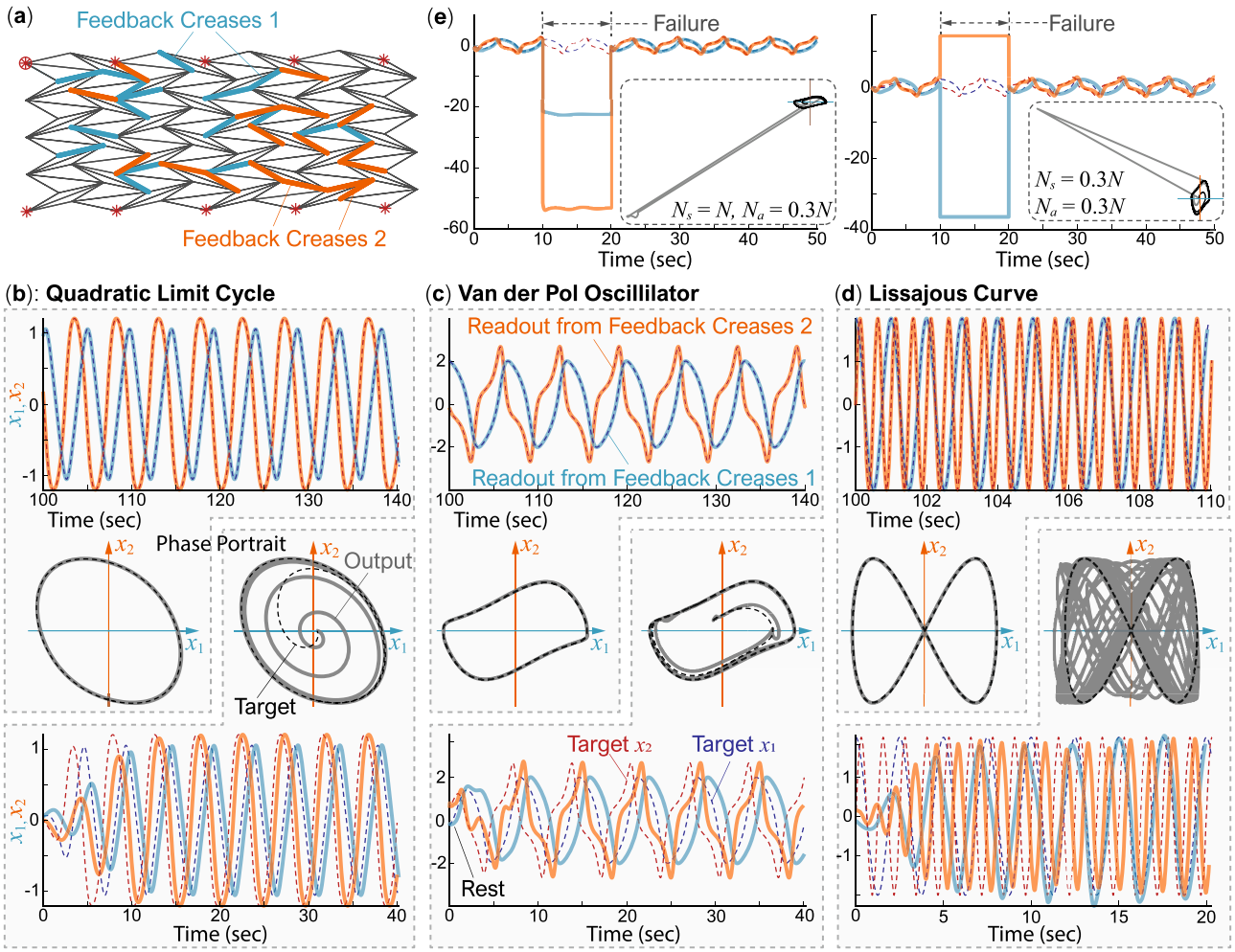

Figure 5. Stable pattern generation under closed-loop using the Miura-ori reservoir. (**a**) This task's origami reservoir includes two groups of feedback creases required to generate 2D limit cycles. (**b**–**d**) The closed-loop trajectories of quadratic limit cycle, Van der Pol oscillator, and the Lissajous curve, respectively. In these plots, the first row of time responses shows the closed-loop output after 100s of training. The third row of time responses shows how the trained reservoir can recover the targeted limit cycles from an initial resting condition. The corresponding phase portraits are as shown in the second row. Here, the dashed curves are targeted trajectories, and the solid curves are the reservoir's outputs. (**e**) Van der Pol limit cycle recovery after the temporary failure of sensor and actuator creases. The two simulations are the same except for the number of sensor creases ($N_s = N$ for the first test, $N_s = 0.3N$ for the second). The insert figures show the corresponding phase-portraits.

$$\dot{x}_2 = -2x_1 + x_2 - x_2\left(x_1^2 + x_2^2\right), \tag{19}$$

where the parameter $\epsilon(t)$ determines the shape of the limit cycle ($\epsilon(t) = 1$ in this case). The Van der Pol oscillator is defined by:

$$\dot{x}_1 = x_2, \tag{20}$$

$$\dot{x}_2 = -x_1 + \left(1 - x_1^2\right)x_2. \tag{21}$$

The Lissajous curve is a graph of two sinusoidal signals parameterized by their frequency ratio ($f_1/f_2 = 0.5$) and phase difference ($\delta = \pi/2$):

$$x_1 = \sin\left(f_1 t + \delta\right) \tag{22}$$

$$x_2 = \sin\left(f_2 t\right) \tag{23}$$

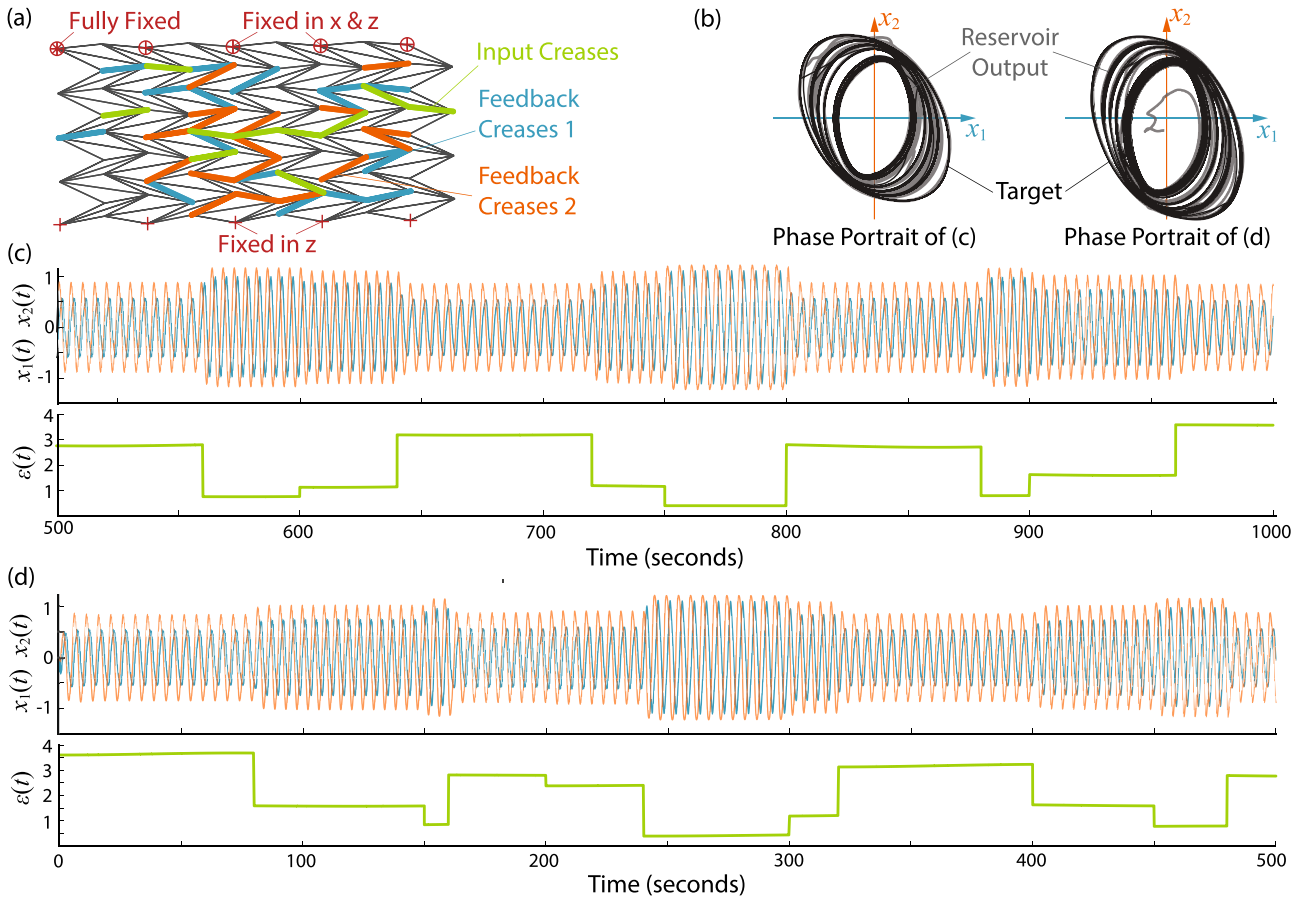

Figure 6. Results of modulation task under closed-loop using The Miura-ori reservoir. (**a**) This task's origami reservoir includes two groups of feedback creases and input creases. (**b**) The phase-portraits, where the targeted trajectories are overlaid on top of the reservoir output. (**c**) Quadratic limit cycle trajectories under closed-loop and the corresponding input signal $\epsilon(t)$. The results are obtained after 500 seconds of training. (**d**) Closed-loop trajectory recovery from the initial resting conditions.

Figure 5b–d show the results of Quadratic limit cycle, Van der Pol oscillator and Lissajous curve generation task, respectively. The origami reservoir can generate all three periodic trajectories just by changing the output readout weights. The MSE for Quadratic limit cycle, Van der Pol oscillator, and Lissajous curves, calculated using the data for first 10 seconds' closed-loop run (M = 10000), are 3.28×10^{-7}, 2.03×10^{-5}, and 5.5×10^{-4}, respectively. As expected, MSE increases as the complexity of the curve increases.

Stability and robustness of the pattern generation. After finding the readout weights, we test the stability of these three limit cycles by starting the origami reservoir from total rest in the close-loop and running it for more than 1000 seconds. The limit cycle is stable if and only it can recover the pattern from zero initial conditions and stays on target for at least 1000 seconds of simulation[19,32]. The results in Fig. 5 indicate that the torsional moments generated from the feedback signals on the feedback creases are sufficient to recover and maintain the three limit cycles from total rest. Small phase differences occur between generated trajectories and the targets because the reservoir takes a slightly different path than the target, and the Lissajous curve takes more than 15 seconds to recover fully. Nonetheless, the origami reservoir successfully passes this test.

To further analyze the robustness of reservoir-generated limit cycles, we simulate actuator and sensor failures. As the origami reservoir generates the Van der Pol limit cycles in these tests, all feedback and sensor creases stop working (aka. their signals set to zero) for 10 seconds. We conduct these tests when all creases are used as sensor creases ($N_s = N$) and when only feedback creases are sensor creases ($N_s = N_a = 0.3N$). The simulation results in Fig. 5e show that, although the reservoir diverges to a trajectory far away from the target during the actuator and sensor failure, it can immediately recover the Van der Pol limit cycles after the end of these failures.

Output modulation task. Output modulation capability allows the reservoir to adjust its output according to a randomly varying input signal without changing the readout weights. This ability is also essential for soft robotic control applications because it allows the robot to switch behaviors according to external stimuli or environmental changes. In this task, we randomly select input creases, which account for 15% of the total creases, in addition to the feedback creases (Fig. 6a). Moreover, all creases are used as sensor creases ($N_s = N$). The simulation results in Fig. 6b shows the generated quadratic limit cycles with modulated input (Eqs. (18, 19)). We are

Parameter	Base value	Distribution
Nodal mass (g)	7	[1,50]
Geometric	Standard	$\sigma = \chi \exp(\frac{-\|\|(N_i - N_j)\|\|}{l})$
imperfections	Miura-ori	$\mu = 0, \chi = 0.4a, l = 4a$
Truss torsional	$k_{b,a} = 1$,	$k_{b,a} = 1$,
stiffness N/(m-rad)	$k_b = 0.2525$	$k_b \in [0.005, 0.5]$

Table 3. Variables for reservoir size and material properties parametric study.

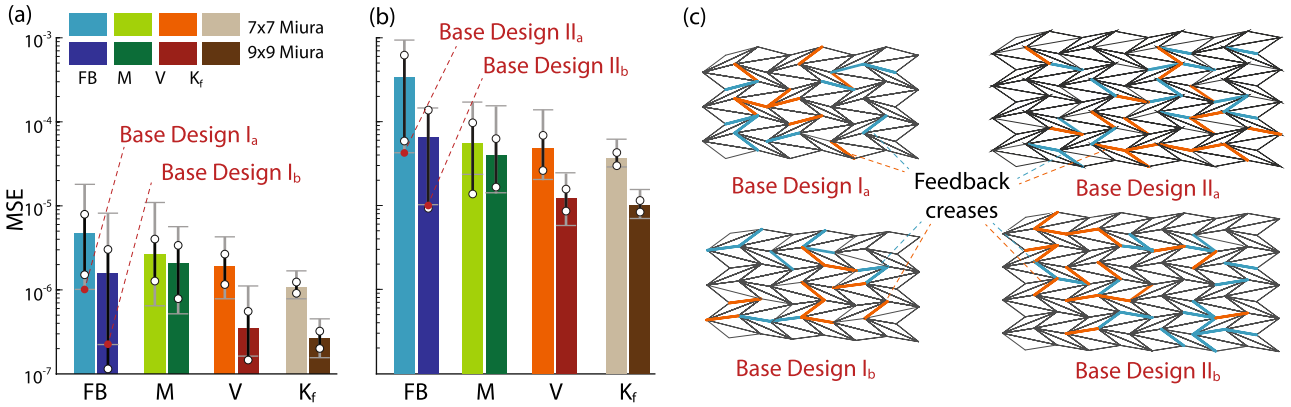

Figure 7. Effect of reservoir size and material properties on the reservoir computing performance. (**a**) The distribution of MSE from the Quadratic limit cycle simulations using random feedback crease distributions and different design parameter distributions. Here "FB" stands for feedback crease distribution, "M" stands for nodal mass distribution, "V" stands for origami vertices geometry perturbation, and "K_f" stands for crease torsional stiffness distribution. It is worth emphasizing that the "FB" results come from one parametric study of 72 unique designs, and the "M," "V," and "K_f" are results of the subsequent simulation. The bar charts depict the average value, standard deviation (circles), and extreme values (horizontal bars) of MSE. (**b**) A similar result from the Van der Pol limit cycle generation task. (**c**) The feedback crease distributions of the four different baseline designs used in this parametric study.

also able to recover the closed-loop trajectory from the initial resting conditions, as shown in Fig. 6c. The phase portraits for the two tasks are depicted in Fig. 6d. The origami reservoir can react to the input and modulate the magnitude of the quadratic limit cycles. The MSE is 3.8×10^{-4}, which is remarkably small, considering this task's complexity.

Correlating physical design and computing performance

In this section, we use the mean squared error (MSE) as the metric to examine the connections between the origami reservoir's design and computing performance. In particular, This analysis aims to investigate the sensitivity of MSE to different parameter changes and identify the optimal origami designs. To this end, in-depth parametric analyses are conducted to examine the effect of (1) reservoir size and material properties, (2) crease pattern geometry, and (3) feedback and sensor crease distribution. We use both Van der Pol and quadratic limit cycle generation tasks to ensure the broad applicability of parametric study results.

Reservoir size, material properties, and vertices perturbation. Figure 7a,b show the results of parametric analyses for Quadratic limit cycle and Van der Pol oscillator limit cycle generation, respectively. We observe that feedback crease distribution affects reservoir computing performance quite significantly. In particular, poorly distributed feedback creases might result in failed pattern generating tasks. Therefore, we first conduct numerical simulations by randomly changing the feedback crease distributions (72 unique designs in total) and identifying the best performing one (with the least MSE). We refer to this best performing feedback crease distribution as the *base design* (Fig. 7c) for the following parametric studies. Then, we conduct another parametric study regarding the nodal mass, crease stiffness, and vertices perturbation. We vary these three parameters, one at a time, for 72 randomly selected designs (six batches of jobs in parallel on a computer with 12 cores). The baseline values and range of the parameters are in Table 3.

The origami reservoir performance turns out to be highly sensitive to the nodal mass variation. As opposed to the uniform nodal mass in base design, a randomly distributed nodal mass can significantly increase or decrease the MSE for both pattern generation tasks. However, randomly distributing mass in an origami sheet is quite challenging in practical applications. So the use of varying mass distribution should be judicially done based on

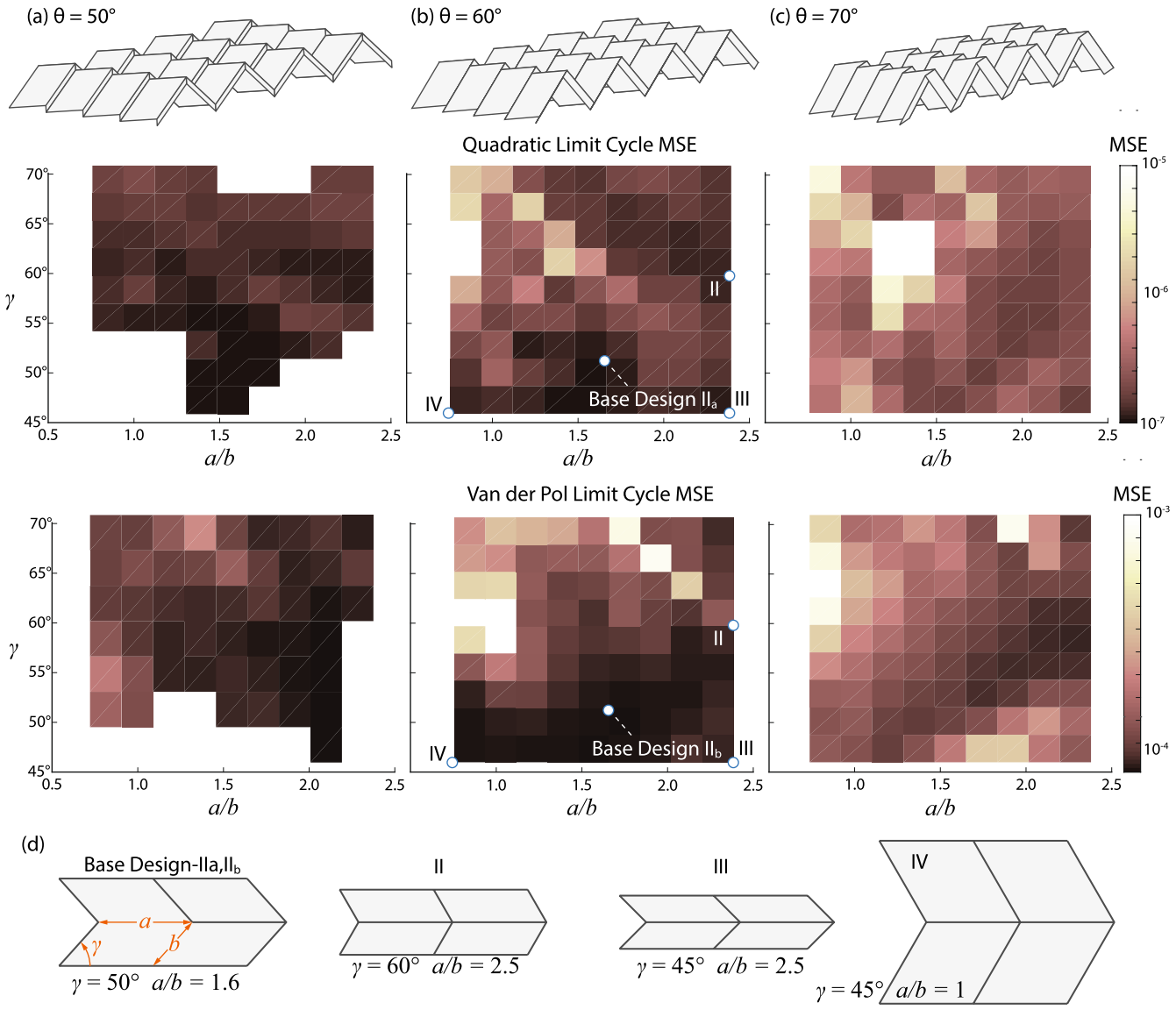

Figure 8. Effect of Miura-ori geometric design on the reservoir performance. (**a**–**c**) The Miura-ori geometry and the corresponding landscape of MSE distribution when $\theta = 50°, 60°$, and 70°, respectively for Quadratic and Van der Pol limit cycle generation. The lighter and darker regions correspond to larger and smaller errors, respectively, while the white regions depict origami designs that failed the computing task. (**d**) The unit cell geometry of four representative designs with the same crease a length but different sector angles γ and crease length ratios a/b.

the particular application at hand. On the other hand, the origami performance is much less sensitive to the crease torsional stiffness. By randomly changing the stiffness, one can achieve performance at par with the base design.

Moreover, we investigate the effects of random geometric imperfection in the base designs of origami reservoir. To this end, we adopt the formulation introduced by Liu et al.[74], which introduce small perturbations to the nodal positions in folded origami. Such imperfections are inevitable in practice due to various manufacturing defects. It is found that these small imperfections do not worsen the MSE significantly and in fact could reduce the MSE by a moderate degree.

It is also worth noting that the larger 9 × 9 Miura origami reservoir performs better than the smaller one because larger origami contains more folding angles to constitute the reservoir state matrix. Therefore, the high-dimensionality of a reservoir is desirable to produce smaller MSE.

Origami design. A unique advantage of origami based structures and materials is their considerable freedom to tailor the geometric design. To this end, we start from the Base Design IIa and IIb of 9 × 9 Miura-ori reservoir, vary its crease length ratio (a/b) and internal sector angle (γ), and then run the quadratic limit cycle and Van der Pol limit cycle tasks with 100 crease length and sector angle combinations at three folding angles. The results of the parametric analysis for $\theta = 50°$, $\theta = 60°$,and $\theta = 70°$ are shown in Fig. 8a–c, respectively. We observe that, at lower folding angles (flatter origami), the origami reservoir has a higher possibility to fail the

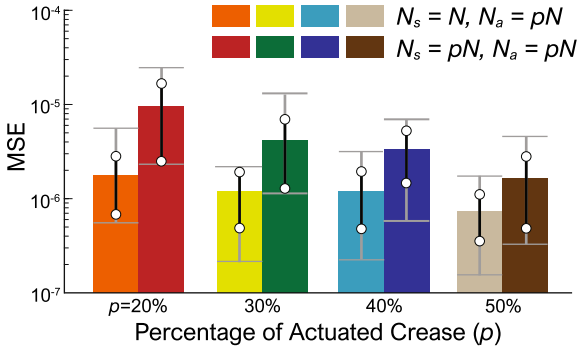

Figure 9. Effect of varying the number of actuator and sensor creases.

pattern generation tasks. The computing performance improves significantly with a reduced MSE as the origami folds more (or as θ increases). This trend is probably because highly folded origami offers an increased range of folding motion.

Moreover, there are two design sets with the lowest MSE: $a/b \approx 1.5, \gamma \approx 45°$, and $a/b \approx 2.5, \gamma \approx 60°$. Figure 8d shows unit cell geometry of four representative designs in this range. Generally speaking, a moderate to high crease-length ratio and small sector angles can create "skewed" origami patterns that appear to give better computing performance across all values of folding angles. The best designs here have MSEs at the order of 10^{-7}, which is of the same magnitude as we found previously by tailoring the nodal mass and crease stiffness.

Actuator and sensors distribution. Finally, it is important, for practical applications, to find the minimum amount of input/feedback and sensor creases required for achieving acceptable computing performance. To this end, we start with the 9×9 Miura-ori reservoir and conduct two tests. In the first test, we vary the percentage of feedback creases ($N_a = 0.2N, 0.3N, 0.4N, 0.5N$, each with 24 randomly generated crease distributions) while using all crease dihedral angles to constitute the reservoir state matrix (i.e., $N_s = N$). In the second test, we use the same feedback crease design and only use these feedback creases' dihedral angles to formulate the reservoir state matrix (i.e., $N_s = N_a$).

We find that if only 20% of crease are used for feedback, the origami reservoir might fail the quadratic limit cycle task. On the other hand, the MSE reduces only marginally as we increase the percentage of feedback creases beyond 30% (Fig. 9). Therefore, we can conclude that using only 30–40% of total creases as the feedback and sensors crease will provide us an adequate computing performance. This result is significant because it shows that, even though a large size (high-dimensionality) of the reservoir is essential for computing performance, one does not need to measure (readout) every reservoir state. In this way, the practical implementation of the origami reservoir can be significantly simplified.

In conclusion, the parametric analyses lay out the strategy to optimize the origami reservoir performance by tailoring the underlying physical and computational design. A larger origami with a higher-dimension can ensure low computational error, but one only needs to use 30% 40% of its creases as the feedback and sensor creases to tap into the origami's computing capacity. Meanwhile, the distribution of these feedback and sensor creases must be carefully chosen with extensive simulations. To further improve computing performance, one can tailor the origami's mass distribution, crease stiffness, and geometric design. Among these options, optimizing the folding geometry should be the most effective because it is easy to implement in practical applications.

Application to soft robotic crawling

This section demonstrates the application of origami reservoir computing to generate an earthworm-inspired peristaltic crawling gait in a robotic system. The earthworm uses peristalsis to navigate uneven terrain, burrow through soil, and move in confined spaces. The lack of complex external appendages (aka., legs or wings) makes earthworm-inspired robots ideal for field exploration, disaster relief, or tunnel drilling[61,76,77]. The body of an earthworm consists of segments (metamerism) grouped into multiple "driving modules"[64,78]. Each driving module includes contracting, anchoring, and extending segments actuated by antagonistic muscles (Fig. 10a). During peristaltic locomotion, these segments alternately contract, anchor (to the environment with the help of *setae*), and extend to create a propagating peristalsis wave, thus moving the body forward.

We design an earthworm-inspired origami robot consisting of two 3×9 Miura-ori sheets connected via a stiff central bridge (Fig. 10b). The left and right half of the robots are symmetric in design, and the central bridge design allows differential motion between the two halves to facilitate turning in response to the external input. In each half of the origami, we embed two groups of feedback creases (Fig. 10b) with feedback weights assigned such that their values for the front and back-half are equal but opposite to each other. This arrangement reduces the number of reference outputs needed to generate a crawling gait. To create a peristalsis locomotion gait, we train the origami reservoir to generate multiple harmonic signals with a phase difference of $\pi/2$ among them (aka. a pattern generation task). We train the robot for 100 seconds and discard the first 15 seconds of data as the washout step.

The ground reactions are modeled by setting the vertical velocity of masses (i.e. point contact with node) touching the ground to zero and the horizontal friction coefficient to be infinite. Inspired from the setae in

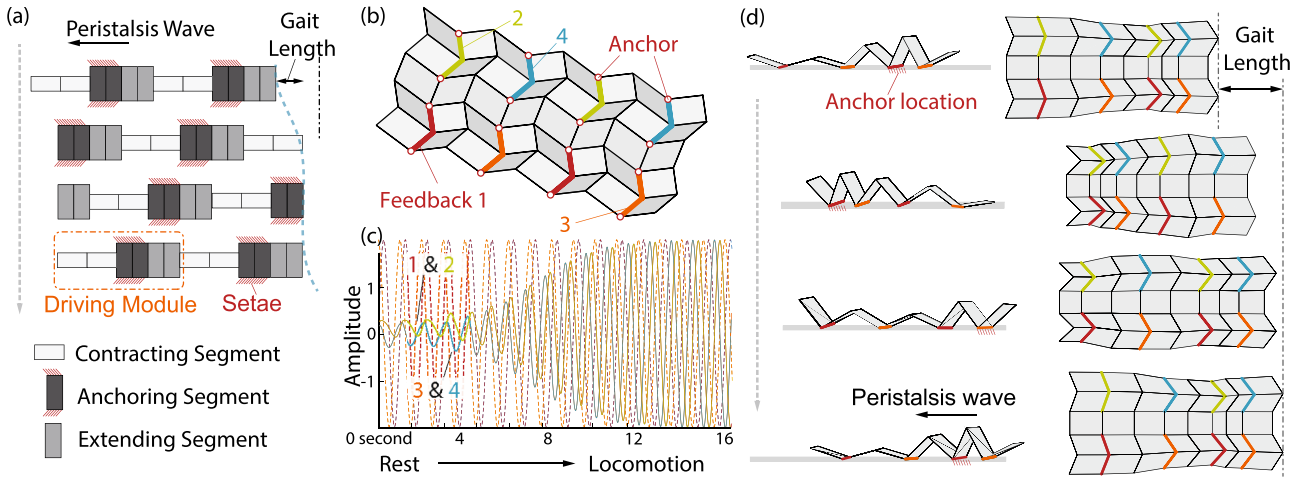

Figure 10. Reservoir computing powered crawling origami robot. (**a**) The kinematics of a peristaltic locomotion cycle in an earthworm. For clarity, the earthworm body is simplified and consists of six identical segments organized into two driving modules. The earthworm body moves forward while the peristaltic wave of anchoring segments (or driving modules) propagates backward. (**b**) The design of an earthworm inspired origami crawling robot that features two stripes of Miura-ori connected by a zig-zag shaped "ridge." This robot has four groups of feedback creases. (**c**) The closed-loop trajectory generated by the feedback creases after training (shown in solid lines) superimposed with the targetted trajectory (shown in dashed lines). (**d**) Peristaltic locomotion cycle in the origami robot as a result of the generated trajectory.

earthworm, we assume presence of passive foldable anchors that lose/grip anchoring to the ground periodically. We apply ideal anchors to the bottom origami creases that are in contact with the surface below. These anchors are assumed to be kinematically attached to the ground when the associated origami crease folds and relaxed as the crease unfolds (or flattens). Such anchor design is feasible by leveraging the origami facets' folding motion, as shown in the author's previous study[64]. This is a hard constraint that can impact the nature and the performance of locomotion. However, it simplifies the study of the body influence by assuming perfect friction conditions in every simulation.

Figure 10c depicts the closed-loop response and the limit cycle recovery from total rest (MSE is 3.9×10^{-04}). Figure 10d (and Supplemental Video A) illustrates the robotic locomotion generated by reservoir computing. As the origami reservoir generates the multiple harmonic signals with a phase difference, its folding motion naturally "synchronizes" to these signals, generating a peristaltic wave of folding and unfolding. As a result, the robot crawls forward like an earthworm, without using any traditional controllers.

Summary and conclusion

We demonstrate the physical reservoir computing capability of origami via extensive benchmark simulations and parametric studies. First, we develop a simulation environment to study the nonlinear origami dynamics and detail the origami reservoir setup. This reservoir successfully achieves many computing tasks such as emulation, pattern generation, and modulation, all of which are relevant to robotic applications. We also conduct comprehensive parametric analysis to uncover the linkage between origami reservoir design and its computing performance. This new knowledge base offers us a guideline to optimize computing performance. To the authors' best knowledge, this is the first study to rigorously examine the performance of physical reservoir computer from the lens of the physical design. Finally, we demonstrate how to embed reservoir computing into an origami robot for control without traditional controllers through the example of peristaltic crawling.

We list four requirements for a mechanical system to be a reservoir in the introduction, and origami satisfies all these requirements. The tessellated origami structures are inherently high-dimensional. For example, a 7×7 Miura-ori with 49 nodes contains $N = 60$ crease dihedral angles, all or a small portion of them can serve as the reservoir states. The nonlinearity of origami partly originates from the nonlinear kinematic relationships between these crease angles and external geometry. Also, since origami patterns are highly structured (ordered), small perturbations in the material properties, imperfections of crease geometry, and the introduction of local actuation are sufficient to destroy the regularity and create disorder. These properties make origami highly nonlinear dynamic reservoirs. The origami reservoir's performance in the emulation task proves that it can act as a nonlinear filter and satisfies fading memory property. Nonlinear patterns can be embedded into the origami reservoir, and the resulting pattern generation is robust against external disturbances and recoverable under different initial conditions, proving separation property. Finally, adding the feedback can create persistent memory, which is conducive to learning new tasks.

For future robots to work autonomously in unstructured and dynamic environments, the robot body and brain have to work together by continuously exchanging information about the current condition, processing this information, and taking appropriate actions. The physical reservoir computing embodied robot shown in this study presents a step toward this vision. The reservoir embedded in the robot body directly gathers information from the distributed sensor-actuator network to perform low-level control tasks like locomotion generation.

Parameter	Value
Reservoir size and material properties	
Size	9×9
Mass of the reservoir	20 g
Geometric design of Miura-ori	
a	31 mm
b	39.2 mm
γ	50.73°
θ	60°
Actuator and sensor creases	
No. of total nodes (N)	81
No. of sensor nodes (N_s)	27

Table 4. Design of the origami reservoir in the emulation task experiment.

The resulting soft robot can generate the global target behavior autonomously without controlling every element individually. Moreover, the generated trajectories could be robust against external disturbances and modulated according to changing working conditions.

A challenge in implementing physical reservoir computing is that many sensors and actuators are required, even though these sensors and actuators can be simple individually. Our results contribute in this regard by showing that only a small portion of origami creases are required to be equipped with sensors and actuators to tap into the reservoir computing power. The essential advantage of an origami reservoir compared to a typical neural reservoir is that it can serve as the physical body of the robot and at the same time perform nonlinear computation for control tasks. Meanwhile, it will be imprudent to directly compare origami reservoirs' computing performance with other physical reservoirs. Only a handful of studies examined the use of soft physical bodies for reservoir computing (e.g., soft silicone arms, tensegrity structures, compliant spine robot). And these systems are often constrained by other non-computing-related requirements. Thus, a comparative study between different physical reservoirs is a worthy topic for future research.

In summary, origami reservoir computing provides an attractive pathway for facilitating synergistic collaboration between the soft robot's body and the brain. The reservoir computing, coupled with unique mechanical properties that origami can offer—multi-stability[51,53,79], nonlinear stiffness[48,49,51,53], and negative Poisson's ratio[48,51,53]—opens up new avenues to the next generation of soft robots with embedded mechanical intelligence.

Methods and materials

We demonstrate the origami reservoir's computational power through a simple experiment of the emulation task described earlier. The experimental setup consists of an APS Dynamics vibration exciter (shaker), power amplifier, National Instruments DAQ, a laptop to generate input command, and a camera to capture the vertical vertices displacements. Detailed origami reservoir design parameters are in Table 4. The reservoir is placed vertically on the vibration exciter and fixed rigidly to its base plate, as shown in Fig. 4a. The external input ($u(t)$) is provided to the base of origami. To experimentally demonstrate the origami reservoir's computing power without unnecessary hardware complexities, we use the vertical (y-direction) displacement of origami nodes as the reservoir state vector. We capture the deformation of origami through a slow-motion video camera at 480 fps with 720p resolution. The nodal displacement in the y-direction is measured through post-processing of the video data in MATLAB. The training procedures for the emulation task remain the same. The results in Fig. 4 clearly show that the physical origami reservoir can indeed emulate the three non-linear systems described in Table 2.

It is worth emphasizing that using a single base excitation as external input and the vertical nodal displacements as the reservoir states only taps into a part of the origami's high dimensional dynamics. Therefore, the mean square error in the experiment is more significant than those with embedded sensors and actuators (Fig. 3). Distributing sensors and actuators on the origami creases will provide better control over the shape of origami, as shown in the example of soft robotic crawling. Nevertheless, this experiment serves as physical proof that the Miura-ori based reservoir can perform complex computational tasks.

Received: 5 February 2021; Accepted: 4 June 2021
Published online: 21 June 2021

References

1. Miriyev, A. & Kovač, M. Skills for physical artificial intelligence. *Nat. Mach. Intell.* **2**, 658–660. https://doi.org/10.1038/s42256-020-00258-y (2020).
2. Laschi, C., Mazzolai, B. & Cianchetti, M. Soft robotics: Technologies and systems pushing the boundaries of robot abilities. *Sci. Robot.* **1**, eaah3690. https://doi.org/10.1126/scirobotics.aah3690 (2016).
3. Cianchetti, M., Calisti, M., Margheri, L., Kuba, M. & Laschi, C. Bioinspired locomotion and grasping in water: The soft eight-arm OCTOPUS robot. *Bioinspiration Biomim.* **10**, 035003. https://doi.org/10.1088/1748-3190/10/3/035003 (2015).

4. Hannan, M. W. & Walker, I. D. Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots. *J. Robot. Syst.* **20**, 45–63. https://doi.org/10.1002/rob.10070 (2003).
5. Ma, K. Y., Chirarattananon, P., Fuller, S. B. & Wood, R. J. Controlled flight of a biologically inspired. *Insect-Scale Robot. Sci.* **340**, 603–607. https://doi.org/10.1126/science.1231806 (2013).
6. Joshi, A., Kulkarni, A. & Tadesse, Y. Fludojelly: experimental study on jellyfish-like soft robot enabled by soft pneumatic composite (spc). *Robotics* **8**, 56. https://doi.org/10.3390/robotics8030056 (2019).
7. Ren, Z., Hu, W., Dong, X. & Sitti, M. Multi-functional soft-bodied jellyfish-like swimming. *Nat. Commun.*. https://doi.org/10.1038/s41467-019-10549-7 (2019).
8. Katzschmann, R. K., Marchese, A. D. & Rus, D. Hydraulic autonomous soft robotic fish for 3D swimming. *Springer Tracts Adv. Robot.* **109**, 405–420. https://doi.org/10.1007/978-3-319-23778-7_27 (2016).
9. Rafsanjani, A., Zhang, Y., Liu, B., Rubinstein, S. M. & Bertoldi, K. Kirigami skins make a simple soft actuator crawl. *Sci. Robot.* **3**, 1–8. https://doi.org/10.1126/scirobotics.aar7555 (2018).
10. Wu, Y. *et al.* Insect-scale fast moving and ultrarobust soft robot. *Sci. Robot.* **4**, eaax1594. https://doi.org/10.1126/scirobotics.aax1594 (2019).
11. Seok, S. *et al.* Meshworm: A peristaltic soft robot with antagonistic nickel titanium coil actuators. *IEEE/ASME Trans. Mechatron.* **18**, 1485–1497. https://doi.org/10.1109/TMECH.2012.2204070 (2013).
12. Hoffmann, M. & Müller, V. C. Simple or complex bodies? Trade-offs in exploiting body morphology for control. *Stud. Appl. Philos. Epistemol. Ration. Ethics* **28**, 335–345. https://doi.org/10.1007/978-3-319-43784-2_17 (2017).
13. Laschi, C. & Mazzolai, B. Lessons from animals and plants: The symbiosis of morphological computation and soft robotics. *IEEE Robot. Autom. Mag.* **23**, 107–114. https://doi.org/10.1109/MRA.2016.2582726 (2016).
14. Trivedi, D., Rahn, C. D., Kier, W. M. & Walker, I. D. Soft robotics: Biological inspiration, state of the art, and future research. *Appl. Bionics Biomech.* **5**, 99–117. https://doi.org/10.1080/11762320802557865 (2008).
15. Paul, C. *Investigation of Morphology and Control in Biped Locomotion*. Ph.D. thesis, University of Zurich (2004).
16. Paul, C. Morphological computation. A basis for the analysis of morphology and control requirements. *Robot. Auton. Syst.* **54**, 619–630. https://doi.org/10.1016/j.robot.2006.03.003 (2006).
17. Caluwaerts, K., D'Haene, M., Verstraeten, D. & Schrauwen, B. Locomotion without a brain: Physical reservoir computing in Tensegrity structures. *Artif. Life* **19**, 35–66. https://doi.org/10.1162/ARTL_a_00080 (2013).
18. Hauser, H., Ijspeert, A. J., Füchslin, R. M., Pfeifer, R. & Maass, W. Towards a theoretical foundation for morphological computation with compliant bodies. *Biol. Cybern.* **105**, 355–370. https://doi.org/10.1007/s00422-012-0471-0 (2011).
19. Nakajima, K. *et al.* A soft body as a reservoir: Case studies in a dynamic model of octopus-inspired soft robotic arm. *Front. Comput. Neurosci.* **7**, 1–19. https://doi.org/10.3389/fncom.2013.00091 (2013).
20. Müller, V. C. & Hoffmann, M. What is morphological computation? On how the body contributes to cognition and control. *Artif. Life* **23**, 1–24. https://doi.org/10.1162/ARTL_a_00219. arXiv:1411.7267 (2017).
21. Tanaka, G. *et al.* Recent advances in physical reservoir computing: A review. *Neural Netw.* **115**, 100–123. https://doi.org/10.1016/j.neunet.2019.03.005. arXiv:1808.04962 (2019).
22. Füchslin, R. M. *et al.* Morphological computation and morphological control: Steps toward a formal theory and applications. *Artif. Life* **19**, 9–34. https://doi.org/10.1162/ARTL_a_00079 (2013).
23. Collins, S. H., Wisse, M. & Ruina, A. A three-dimensional walking robot with two legs and knees. *Int. J. Robot. Res.* **20**, 607–615. https://doi.org/10.1177/02783640122067561 (2001).
24. Floreano, D. *et al.* Miniature curved artificial compound eyes. *Proc. Natl. Acad. Sci. U.S.A.* **110**, 9267–9272. https://doi.org/10.1073/pnas.1219068110 (2013).
25. Jaeger, H. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. German National Research Center for Information Technology GMD Technical Report **148**, 13 (2001).
26. Maass, W. W., Markram, H. & Natschläger, T. The "liquid computer": A novel strategy for real-time computing on time series. *Spec. Issue Found. Inf. Process. TELEMATIK* **8**, 39–43. https://doi.org/10.1017/CBO9781107415324.004.arXiv:1011.1669v3 (2002).
27. Maass, W., Joshi, P. & Sontag, E. D. Computational aspects of feedback in neural circuits. *PLoS Comput. Biol.* **3**, 0015–0034. https://doi.org/10.1371/journal.pcbi.0020165 (2007).
28. Maass, W. Liquid state machines: Motivation, theory, and applications. In *Computability in context: computation and logic in the real world*, 275–296 (World Scientific, 2011).
29. Lukoševičius, M. & Jaeger, H. Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.* **3**, 127–149. https://doi.org/10.1016/j.cosrev.2009.03.005 (2009).
30. Schrauwen, B., Verstraeten, D. & Van Campenhout, J. An overview of reservoir computing: Theory, applications and implementations. In *Proceedings of the 15th European Symposium on Artificial Neural Networks*, 471–482 (2007).
31. Nakajima, K. Physical reservoir computing—An introductory perspective. *Jpn. J. Appl. Phys.* **59**. https://doi.org/10.35848/1347-4065/ab8d4f. arXiv:2005.00992 (2020).
32. Hauser, H., Ijspeert, A. J., Füssuchslin, R. M., Pfeifer, R. & Maass, W. The role of feedback in morphological computation with compliant bodies. *Biol. Cybern.* **106**, 595–613. https://doi.org/10.1007/s00422-012-0516-4 (2012).
33. Morales, G. *et al.* Mass-spring damper array as a mechanical medium for computation. *Int. Conf. Artif. Neural Netw.* **1**, 208–217. https://doi.org/10.1007/978-3-030-01424-7 (2018).
34. Caluwaerts, K. & Schrauwen, B. The body as a reservoir: Locomotion and sensing with linear feedback. In *Conference Proceedings: 2nd International Conference on Morphological Computation*, 3 (2011).
35. Li, T., Nakajima, K., Cianchetti, M., Laschi, C. & Pfeifer, R. Behavior switching using reservoir computing for a soft robotic arm. *Proc. IEEE Int. Conf. Robot. Autom.* **1**, 4918–4924. https://doi.org/10.1109/ICRA.2012.6225366 (2012).
36. Nakajima, K., Hauser, H., Li, T. & Pfeifer, R. Exploiting the dynamics of soft materials for machine learning. *Soft Robot.* **5**, 339–347. https://doi.org/10.1089/soro.2017.0075 (2018).
37. Fernando, C. & Sojakka, S. Pattern recognition in a bucket. In *Advances in Artificial Life* (eds Banzhaf, W. *et al.*) 588–597 (Springer, 2003).
38. Degrave, J., Caluwaerts, K., Dambre, J. & Wyffels, F. Developing an embodied gait on a compliant quadrupedal robot. In *IEEE International Conference on Intelligent Robots and Systems* **2015-Decem**, 4486–4491. https://doi.org/10.1109/IROS.2015.7354014 (2015).
39. Agogino, A. K., SunSpiral, V. & Atkinson, D. Super ball bot-structures for planetary landing and exploration. *NASA Technical Report* (2018).
40. Urbain, G., Degrave, J., Carette, B., Dambre, J. & Wyffels, F. Morphological properties of mass-spring networks for optimal locomotion learning. *Front. Neurorobot.* **11**, 1–13. https://doi.org/10.3389/fnbot.2017.00016 (2017).
41. Peraza-Hernandez, E. A., Hartl, D. J., Malak, R. J. Jr. & Lagoudas, D. C. Origami-inspired active structures: A synthesis and review. *Smart Mater. Struct.* **23**, 094001. https://doi.org/10.1088/0964-1726/23/9/094001 (2014).
42. Rus, D. & Tolley, M. T. Design, fabrication and control of origami robots. *Nat. Rev. Mater.* **3**, 1. https://doi.org/10.1038/s41578-018-0009-8 (2018).
43. Ning, X. *et al.* Assembly of advanced materials into 3d functional structures by methods inspired by origami and kirigami: A review. *Adv. Mater. Interfaces* **5**, 1800284. https://doi.org/10.1002/admi.201800284 (2018).

44. Morris, E., McAdams, D. A. & Malak, R. The state of the art of origami-inspired products: A review. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 50169, V05BT07A014. https://doi.org/10.1115/DETC2016-59629 (American Society of Mechanical Engineers, 2016).
45. Filipov, E. T., Paulino, G. H. & Tachi, T. Origami tubes with reconfigurable polygonal cross-sections. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **472**, 20150607. https://doi.org/10.1098/rspa.2015.0607 (2016).
46. Morgan, J., Magleby, S. P. & Howell, L. L. An approach to designing origami-adapted aerospace mechanisms. *J. Mech. Design* **138**, 052301. https://doi.org/10.1115/1.4032973 (2016).
47. Melancon, D., Gorissen, B., Garcia-Mora, C. J., Hoberman, C. & Bertoldi, K. Multistable inflatable origami structures at the meter-scale. *Nature (accepted)*https://doi.org/10.1038/s41586-021-03407-4 *(2021)*.
48. Schenk, M. & Guest, S. D. Geometry of Miura-folded metamaterials. *Proc. Natl. Acad. Sci.* **110**, 3276–3281. https://doi.org/10.1073/pnas.1217998110 (2013).
49. Silverberg, J. L. *et al.* Using origami design principles to fold reprogrammable mechanical metamaterials. *Science* **345**, 647–650. https://doi.org/10.1126/science.1252876 (2014).
50. Yasuda, H., Gopalarethinam, B., Kunimine, T., Tachi, T. & Yang, J. Origami-based cellular structures with in situ transition between collapsible and load-bearing configurations. *Adv. Eng. Mater.* **1900562**, 1900562. https://doi.org/10.1002/adem.201900562 (2019).
51. Li, S., Fang, H., Sadeghi, S., Bhovad, P. & Wang, K. W. Architected origami materials: How folding creates sophisticated mechanical properties. *Adv. Mater.* **31**, 1–18. https://doi.org/10.1002/adma.201805282 (2019).
52. Yan, Z. *et al.* Controlled mechanical buckling for origami-inspired construction of 3D microstructures in advanced materials. *Adv. Funct. Mater.* **26**, 2629–2639. https://doi.org/10.1002/adfm.201504901 (2016).
53. Kamrava, S., Mousanezhad, D., Ebrahimi, H., Ghosh, R. & Vaziri, A. Origami-based cellular metamaterial with auxetic, bistable, and self-locking properties. *Sci. Rep.* **7**, 46046. https://doi.org/10.1038/srep46046 (2017).
54. Miyashita, S. *et al.* Ingestible, controllable, and degradable origami robot for patching stomach wounds. In *Proceedings—IEEE International Conference on Robotics and Automation* **2016-June**, 909–916. https://doi.org/10.1109/ICRA.2016.7487222 (2016).
55. Miyashita, S., Guitron, S., Li, S. & Rus, D. Robotic metamorphosis by origami exoskeletons. *Sci. Robot.* **2**, eaao4369. https://doi.org/10.1126/scirobotics.aao4369 (2017).
56. Belke, C. H. & Paik, J. Mori: A modular origami robot. *IEEE/ASME Trans. Mechatron.* **22**, 2153–2164. https://doi.org/10.1109/TMECH.2017.2697310 (2017).
57. Onal, C. D., Tolley, M. T., Wood, R. J. & Rus, D. Origami-inspired printed robots. *IEEE/ASME Trans. Mechatron.* **20**, 2214–2221. https://doi.org/10.1109/TMECH.2014.2369854 (2015).
58. Onal, C. D., Wood, R. J. & Rus, D. An origami-inspired approach to worm robots. *IEEE/ASME Trans. Mechatron.* **18**, 430–438. https://doi.org/10.1109/TMECH.2012.2210239 (2013).
59. Yan, R. *et al.* OriSnake: Design, fabrication and experimental analysis of a 3-D origami snake robot. *IEEE Robot. Autom. Lett.* **3**, 1. https://doi.org/10.1109/LRA.2018.2800112 (2018).
60. Novelino, L. S., Ze, Q., Wu, S., Paulino, G. H. & Zhao, R. Untethered control of functional origami microrobots with distributed actuation. *Proc. Natl. Acad. Sci.* **117**, 24096–24101. https://doi.org/10.1073/pnas.2013292117 (2020).
61. Fang, H., Zhang, Y. & Wang, K. W. Origami-based earthworm-like locomotion robots. *Bioinspiration Biomim.* **12**, 0665003. https://doi.org/10.1088/1748-3190/aa8448 (2017).
62. Jeong, D. & Lee, K. Design and analysis of an origami-based three-finger manipulator. *Robotica*. https://doi.org/10.1017/S0263574717000340 (2017).
63. Kaufmann, J., Bhovad, P. & Li, S. Harnessing the multistability of kresling origami for reconfigurable articulation in soft robotic arms. *Soft Robot.* **00**, soro.2020.0075. https://doi.org/10.1089/soro.2020.0075. arXiv:2008.07421 (2021).
64. Bhovad, P., Kaufmann, J. & Li, S. Peristaltic locomotion without digital controllers: Exploiting the origami multi-stability to coordinate robotic motions. *Extreme Mech. Lett.* **32**, 100552. https://doi.org/10.1016/j.eml.2019.100552. arXiv:1906.04091 (2019).
65. Zhakypov, Z., Falahi, M., Shah, M. & Paik, J. The design and control of the multi-modal locomotion origami robot, tribot. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4349–4355. https://doi.org/10.1109/IROS.2015.7353994 (2015).
66. Sadeghi, S., Allison, S., Betsill, B. & Li, S. TMP origami jumping mechanism with nonlinear stiffness. *Smart Mater. Struct.*https://doi.org/10.1088/1361-665X/abf5b2 *(2021)*.
67. Legenstein, R. A. & Maass, W. *What Makes a Dynamical System Computationally Powerful?* 1st edn, 127–154 (MIT Press, 2007).
68. Schenk, M. & Guest, S. D. Origami folding: A structural engineering approach. In *Proceedings of The Fifth International Meeting of Origami Science Mathematics and Education (5OSME)*, 291–303 (2011).
69. Liu, K. & Paulino, G. H. Nonlinear mechanics of non-rigid origami: An efficient computational approach. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **473**, 20170348. https://doi.org/10.1098/rspa.2017.0348 (2017).
70. Ghassaei, A., Demaine, E. D. & Gershenfeld, N. Fast, interactive origami simulation using GPU computation. *Origami* **7**, 1151–1166 (2018).
71. Silverberg, J. L. *et al.* Origami structures with a critical transition to bistability arising from hidden degrees of freedom. *Nat. Mater.* **14**, 389–393. https://doi.org/10.1038/nmat4232 (2015).
72. Hiller, J. & Lipson, H. Dynamic simulation of soft multimaterial 3D-printed objects. *Soft Robot.* **1**, 88–101. https://doi.org/10.1089/soro.2013.0010 (2014).
73. Paul, C., Valero-Cuevas, F. J. & Lipson, H. Design and control of tensegrity robots for locomotion. *IEEE Trans. Robot.* **22**, 944–957. https://doi.org/10.1109/TRO.2006.878980 (2006).
74. Liu, K., Novelino, L. S., Gardoni, P. & Paulino, G. H. Big influence of small random imperfections in origami-based metamaterials. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **476**, 20200236. https://doi.org/10.1098/rspa.2020.0236 (2020).
75. Atiya, A. F. & Parlos, A. G. New results on recurrent network training: Unifying the algorithms and accelerating convergence. *IEEE Trans. Neural Netw.* **11**, 697–709. https://doi.org/10.1109/72.846741 (2000).
76. Calderón, A. A., Ugalde, J. C., Chang, L., Cristóbal Zagal, J. & Pérez-Arancibia, N. O. An earthworm-inspired soft robot with perceptive artificial skin. *Bioinspiration Biomim.* **14**, 0–12. https://doi.org/10.1088/1748-3190/ab1440 (2019).
77. Kamata, M., Yamazaki, S., Tanise, Y., Yamada, Y. & Nakamura, T. Morphological change in peristaltic crawling motion of a narrow pipe inspection robot inspired by earthworm's locomotion. *Adv. Robot.* **32**, 386–397. https://doi.org/10.1080/01691864.2017.1417158 (2018).
78. Quillin, K. & Quillin. Kinematic scaling of locomotion by hydrostatic animals: Ontogeny of peristaltic crawling by the earthworm lumbricus terrestris. *J. Exp. Biol.* **202**(Pt 6), 661–674 (1999).
79. Waitukaitis, S., Menaut, R., Chen, B. G. & van Hecke, M. Origami multistability: From single vertices to metasheets. *Phys. Rev. Lett.* **114**, 055503. https://doi.org/10.1103/PhysRevLett.114.055503 (2015).

Acknowledgements

The authors acknowledge the support from the National Science Foundation (CMMI-1933124), as well as the Clemson University for the generous allotment of computing time on Palmetto cluster.

Author contributions

All authors conceived the experiments and analyzed the results, P.B. conducted the experiments. All authors reviewed the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-021-92257-1.

Correspondence and requests for materials should be addressed to P.B.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.