SCIENCE CHINA

Information Sciences

Supplementary Materials for

A novel sim-to-real reinforcement learning algorithm for soft growing robot navigation

Haoran Wu[1], Fuchun Sun[2*], Zengxin Kang[1], Lin Tang[1] & Zhongyi Chu[1*]

*Corresponding author. Email: fcsun@tsinghua.edu.cn; chuzy@buaa.edu.cn

1 Environmental Interaction Navigation of Soft Growing Robot

The soft growing robot exhibits high compliance, enabling it to navigate through confined environments via interactions with surrounding obstacles. According to the study by Greer et al.[1]-[2], the soft growing robot can be modeled as an inflatable cantilever beam. A lumped parameter model is developed for the soft growing robot, segmenting it at points of bending or collision with the environment.

$$\begin{aligned} \vec{c}_{i+1} &= l_i \hat{e}_{i+1} + \vec{c}_i \\ \hat{e}_{i+1} &= R_z(\theta_i)\hat{e}_i \end{aligned} \tag{1}$$

Here, l_i represents the length of each segment of the robot, c_i denotes the position of the pivot point, and $R_z(\theta)$ represents a rotation of θ degrees around the Z-axis. In this way, the overall shape of the soft growing robot can be characterized by the length of each segment or the bending angles at each pivot point.

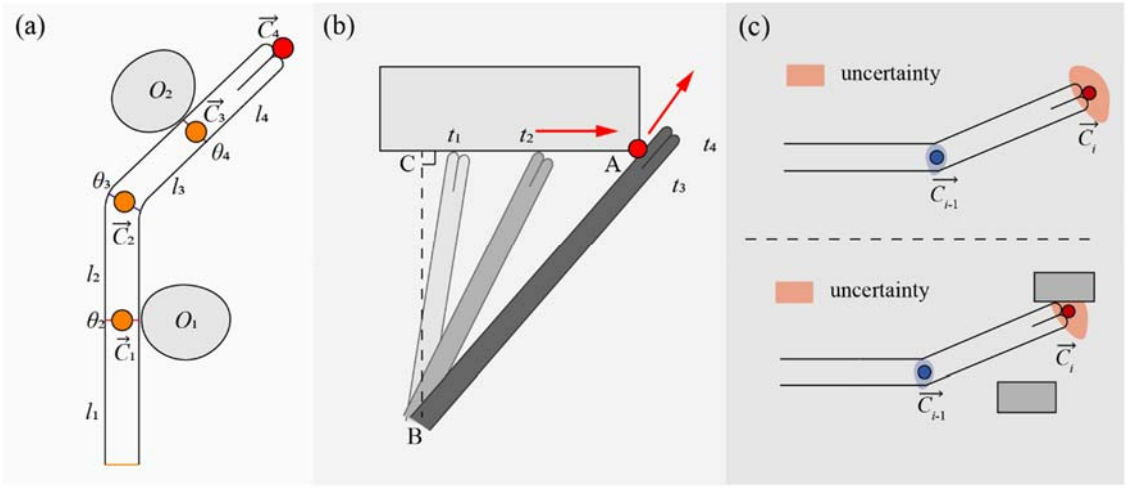

Figure S1 Soft growing robot and its environment-interaction navigation strategy (a) Lumped parameter model of the soft growing robot (b) Condensation points and environmental guidance (c) Uncertainty of the soft growing robot

When the soft growing robot collides with the environment, its growth direction changes, forming an angle of less than 90° with its original trajectory and aligning parallel to one side of the obstacle. Thus, environmental structures serve as both guidance and constraint, effectively shaping the robot's trajectory. As illustrated in Figure S1(b), all robots initialized within the ∠CBA region are steered toward point A due to the environmental layout. The red-shaded area represents the uncertainty in the

robot's tip position, primarily caused by manufacturing tolerances. However, this uncertainty is significantly reduced through physical interaction with environmental boundaries, which constrain the robot's path and guide its growth direction.

2 Control and Modeling of PAMs

The proportional valve used in this study is the VEAB-L-26 model from Festo [3]. The response time of the proportional valve is approximately 50ms. As illustrated in Figure S2, the pneumatic system consists of a positive pressure air source connected to the proportional valve, which is followed by a two-position three-way solenoid valve. The regulated airflow is then directed either to the PAMs or to the body of the soft growing robot.

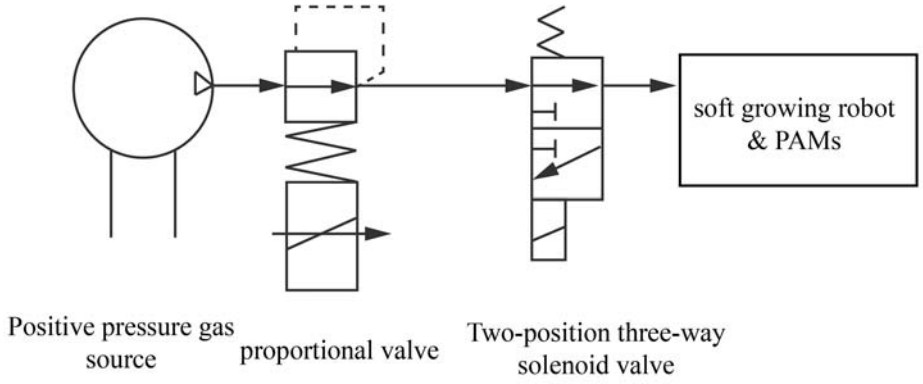

Figure S2. Schematic diagram of the pneumatic system structure

During the fabrication process, one end of the plastic film is first heat-sealed. Subsequently, fishing lines are attached at regular intervals to complete the construction of the PAM. When inflated, the PAM contracts in length, causing the robot to bend toward one side. PAMs are mounted on both the left and right sides of the soft growing robot. To simplify modeling and control, a constant curvature model is employed to describe their behavior.

$$l_i = l - \alpha(d/2) \tag{2}$$

Where l represents the length of the PAM attached to the soft growing robot, l_i denotes the length of each segment of the PAM, and α is the bending angle.

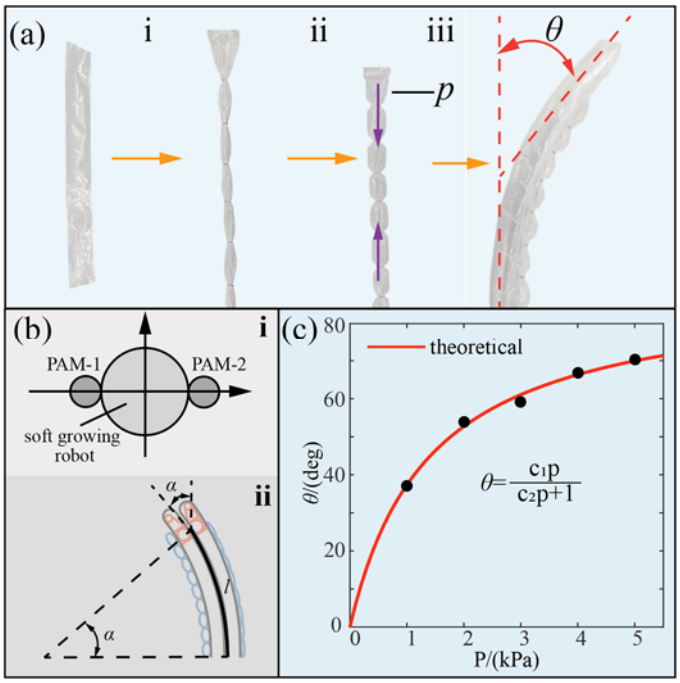

Figure S3. Fabrication and Modelling of PAMs

The internal pressure of the pneumatic artificial muscle and the bending angle at the robot's tip approximately follow the relationship described below.[4]-[5]

$$\kappa = \frac{\theta}{l} = \frac{c_1 p}{c_2 p + 1} \tag{3}$$

where

$$c_1 = \frac{D_{\text{tube}} k_1 \left(l^{eq} - l_1^{eq} \right)}{2 l^{eq} K_t}$$

$$c_2 = \frac{k_1 l_1^{eq} K_t + K k_1 l^{eq} D_{\text{tube}}}{K l^{eq} K_t}$$

In this context, K_1 represents the stretching stiffness of the PAM; K_t and k are the torsional stiffness and stretching stiffness of the soft growing robot, respectively.

In practical applications, the bending angle of the robot is determined using the gyroscope's end pose, and the length and internal pressure of the pneumatic artificial muscles are further calculated. Precise control of the pneumatic artificial muscles is achieved using a proportional valve.

3 Sim-to-Real Simulation Framework for Soft Growing Robot

This section describes the sim-to-real simulation framework [6]-[7] of the soft growing robot from two perspectives: structural characteristics and actuation strategies. The structural characteristics of the soft growing robot are analyzed, simplified, and subsequently modeled in the Unity virtual environment.

The soft growing robot extends its length through the eversion of a membrane; however, accurately simulating this eversion dynamics in the simulation software is challenging, posing difficulties for real-time execution. To address this, we represent the flexible body of the soft growing robot using multiple serially connected rigid capsules and model its growth process by incrementally appending new rigid capsules at the tip.

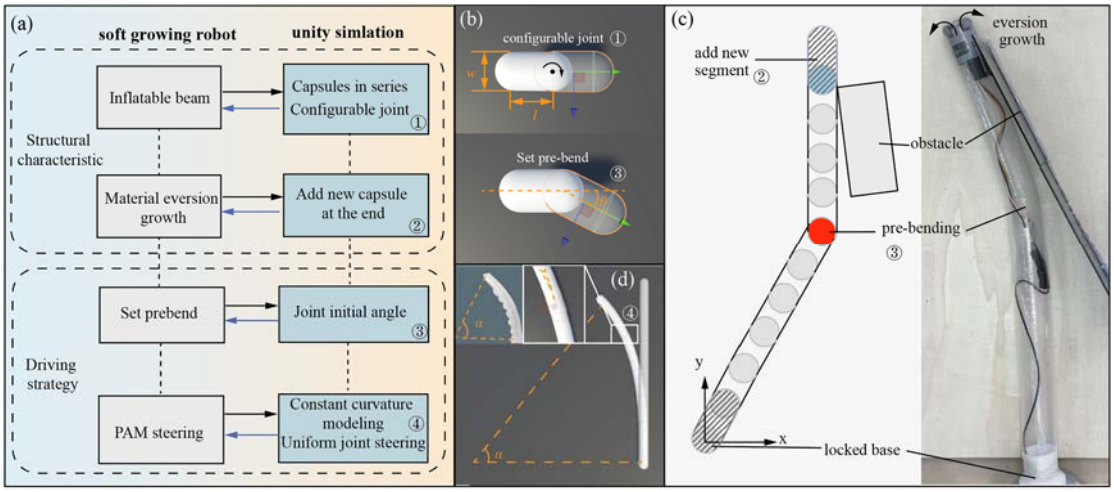

Figure S4 (a) Sim-to-real framework of soft growing robot (b) Schematic of the capsule structure and pre0-bending (c) Schematic of the soft growing robot (d) Schematic of the pneumatic artificial muscle.

As shown in Figure S4, adjacent capsule bodies are connected via configurable joints, where the upper sphere center of one capsule coincides with the lower sphere center of the next, while collisions between adjacent capsules are disabled. The simulation focuses exclusively on the in-plane behavior of the soft growing robot. Therefore, the rotational degrees of freedom about the X and Z axes are constrained, leaving only rotation about the Y-axis unrestricted. To simulate the behavior of an inflatable flexible beam, appropriate springs and dampers are incorporated into the joints. The material properties of the capsule bodies are tuned to introduce a certain degree of elasticity. The first capsule body is immobilized to serve as the base of the

soft growing robot. By adjusting the initial angles of the configurable joints, we can simulate the pre-bent shape of the soft growing robot. Assuming that pneumatic artificial muscles are symmetrically distributed on both sides of the robot, its bending behavior can be approximated using a constant curvature model. At this point, actuators are added to all joints, enabling uniform rotation by a predefined angle to simulate the effect of PAMs.

To closely approximate real-world conditions, domain randomization is applied to the virtual environment, and consistent random seeds are used across Unity and Python-based reinforcement learning code to ensure reproducibility and facilitate debugging. The friction coefficient between the capsule bodies and the ground is randomized within the range of 0.2 to 0.3. Configurable joint parameters are tuned such that angular stiffness varies between 80 and 150, angular damping between 25 and 40, and joint angle limits are constrained within 20° to 30°. To further simulate real-world fabrication uncertainties during the training of the second-layer network, positional deviations are introduced between the centers of adjacent capsules at pre-bent joints, and angular deviations are allowed in the preset bending angles. Additionally, to enhance robustness, simulation parameters are randomized every 20 episodes during training, introducing controlled uncertainty into the learning process.

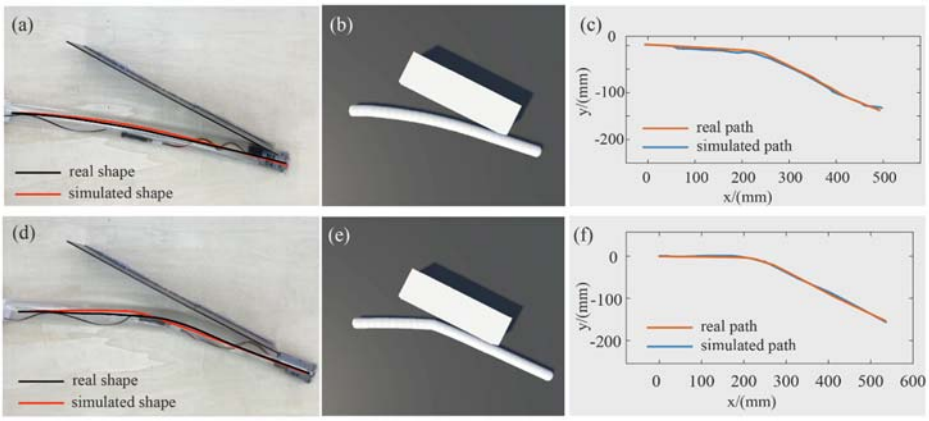

Figure S5 Unity simulation validation (a) Real-world scenario without pre-bending (b) Virtual scenario without pre-bending (c) Comparison of actual and simulated trajectories without pre-bending (d) Real-world scenario with pre-bending (e) Virtual scenario with pre-bending (f) Comparison of actual and simulated trajectories with pre-bending

To verify the effectiveness of the virtual environment simulation, we constructed a simple test setup where the soft growing robot was operated both with and without pre-bending. As illustrated in Figure S5, the trajectory of the end-effector and the overall shape of the robot after movement were compared. When the total length of the soft

growing robot reached approximately 800 mm, the maximum structural deviation was 3.66 mm, and the maximum end-effector trajectory error was 3.74 mm. Therefore, regardless of whether pre-formed bending was applied, the simulation results closely matched the actual outcomes.

4 Dual-Layer DDPG Network

This section presents the design of a dual-layer DDPG network for the control and motion planning of the soft growing robot. Compared to other reinforcement learning algorithms, DDPG is more suitable for high-dimensional continuous control tasks, offering faster sampling efficiency and convergence speed (compared to PPO), higher stability, and lower computational cost (compared to SAC). The first-layer network primarily learns the environmental interaction strategy of the growing robot, determining the pre-bending configuration to reach the target position. The second-layer network, based on the pre-bending action selected by the first-layer network, accounts for manufacturing errors and sensor information to regulate the motion of pneumatic artificial muscles for error compensation.

4.1 Pre-bending Action Selection Network

This section details the action selection strategy, the design of the reward function, and the reinforcement learning training process conducted in both the Unity virtual environment and Python.

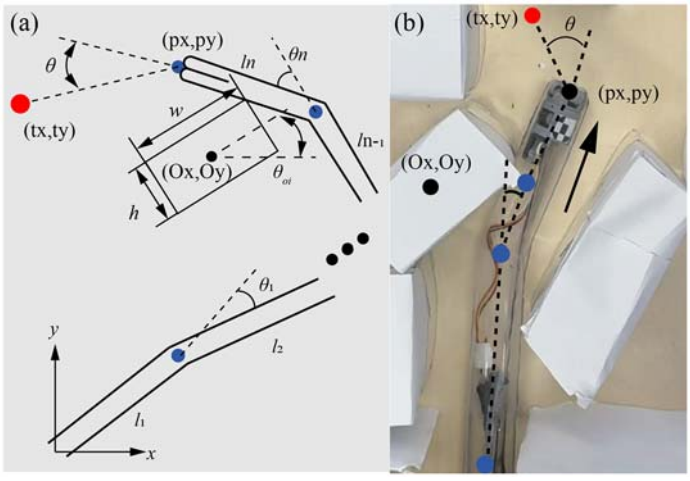

Figure S6 (a) State vector of the soft growing robot (b) Physical prototype demonstration of state variables.

The action of the soft growing robot determines whether to set pre-bending or the angle of pre-bending.

$$a_t = \theta_{\text{bend}}, \begin{cases} \theta_{\text{bend}} < 0, \text{trun left} \\ \theta_{\text{bend}} = 0, \text{straight} \\ \theta_{\text{bend}} > 0, \text{trun right} \end{cases} \tag{4}$$

In practical scenarios, pre-bending cannot be continuously adjusted. Therefore, after the soft growing robot selects a nonzero action, it must choose five consecutive zero actions (i.e., move straight) before selecting another nonzero action.

$$a_t = \begin{cases} 0, c_t < 5 \\ a_t, c_t \geq 5 \end{cases}$$

where

$$c_{t+1} = \begin{cases} c_t + 1, a_t = 0 \\ 0, a_t \neq 0 \end{cases}$$

The reward for the soft growing robot consists of two parts: the reward for approaching the target r_{target} and the reward for turning behavior $_{\text{raction}}$, where w_{action} and w_{target} are their respective weights.

$$r = w_{action} \cdot r_{action} + w_{\text{target}} \cdot r_{\text{target}} \tag{5}$$

where d represents the Euclidean distance between the target point and the robot's tip, while θ_{t} is the angle between the robot's growth direction and the target direction.

$$r_{\text{target}} = e^{-10d}, d = \sqrt{\left(t_x - p_x\right)^2 + \left(t_y - p_y\right)^2} \tag{6}$$

$$\begin{cases} r_{action} = \dfrac{20}{\left(\theta_t - \theta_{\text{bend}} + 1\right)}, \left|\theta_t - \theta_{\text{bend}}\right| < \theta_t \\ r_{action} = -\left|\theta_{\text{bend}}\right|, \left|\theta_t - \theta_{\text{bend}}\right| > \theta_t \end{cases} \tag{7}$$

θ_{bend} is constrained within (-20°,20°) to prevent excessive turning. The target-approaching reward adopts an exponential function to ensure smoother reward transitions. The turning behavior reward is calculated based on the angle between the robot's growth direction before and after turning and the target point, encouraging reasonable turning while penalizing excessive and ineffective turns.

The algorithm flowchart is illustrated in Figure S7, where the orange sections denote the main steps executed in Python, while the blue sections represent the main steps in Unity. In Unity, the current environment information is obtained and transmitted to Python. The reinforcement learning algorithm then selects an action and computes the reward function. The experience is stored based on the computed reward, and the target network is updated accordingly. Next, in Unity, a new capsule is generated based on the received action information, and the system evaluates whether the target has been reached. If the target has not been reached, the process loops back to environment information acquisition. After each single-step simulation, the environment is reset. Once the success rate or training episode surpasses a predefined threshold, the experience replay buffer and reinforcement learning model are stored.

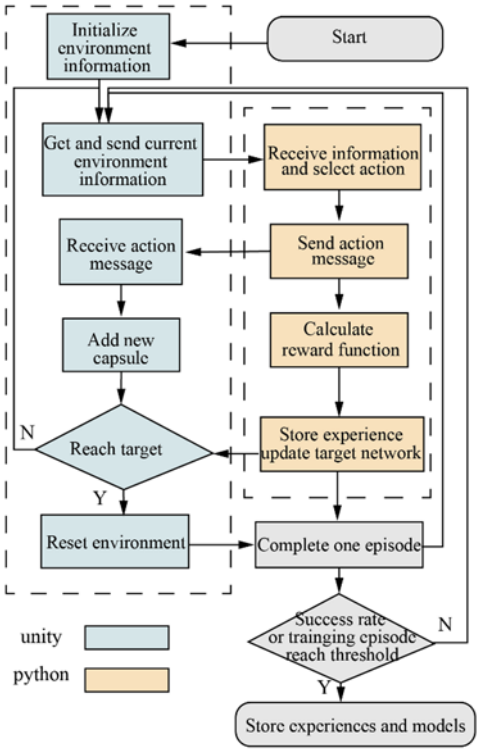

Figure S7: Flowchart of the pre-bending action selection network algorithm for the soft growing robot.

4.2 Perception and Error Correction Network

This section describes the sources of error in the soft growing robot, the reward function design for the perception and error correction network, and the corresponding training process in the virtual environment. During the robot's motion, deviations from

the theoretical trajectory inevitably occur, primarily due to two sources of error: manufacturing inaccuracies in pre-bending and sensor measurement noise.

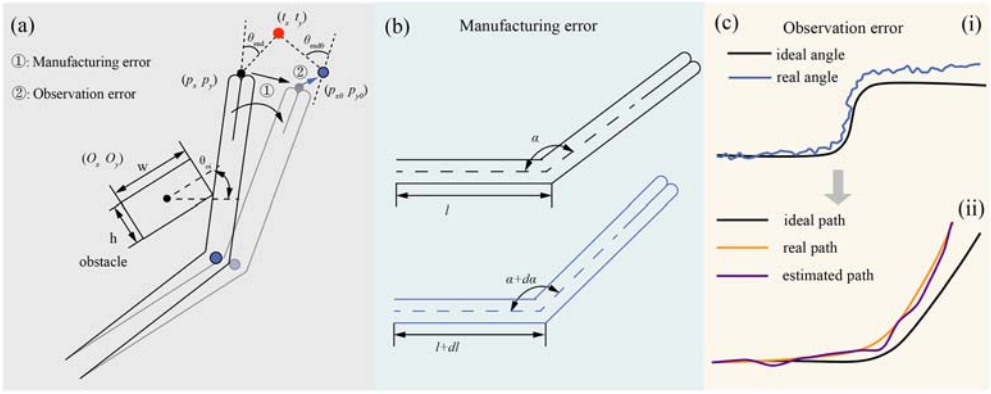

Figure S8. (a) Error sources and environmental variables of the soft growing robot (b) Schematic diagram of manufacturing errors (c) Schematic diagram of observation errors.

When assigning pre-bending configurations to the soft growing robot, deviations often arise between the intended and actual bending angles and positions due to fabrication limitations.(Figure S8) In real-world scenarios, gyroscopes are affected by noise and drift, leading to discrepancies between the ideal and measured trajectories. To simulate this in the virtual environment, we treat the direction of the end capsule as the ground truth for the robot' s orientation, while the measured value θ' is generated by adding Gaussian white noise and a drift term to the theoretical orientation.

$$\theta'(t) = \theta_{end}(t) + n(t) + b(t) \tag{8}$$

Here, $n(t)$ represents zero-mean Gaussian white noise with variance σ_1^2 , which models the high-frequency sensor noise:

$$n(t) \sim N(0, \sigma_1^2)$$

The drift term $b(t)$ models the low-frequency bias in the measurement and follows a random walk process, defined by:

$$b(t) = b(t-1) + w_b(t) \tag{9}$$

where $w_b(t)$ is zero-mean Gaussian noise with variance σ_2^2, representing the incremental drift noise at each time step:

$$w_b(t) \sim N(0, \sigma_2^2)$$

This model effectively captures both the short-term random noise and the long-term drift in the measured orientation.

The reinforcement learning reward is computed based on the discrepancy between the actual and theoretical end positions resulting from PAM actuation. A reward is assigned if the movement reduces the positional error, bringing the end position closer to the target, whereas a penalty is applied if the movement increases the error.

$$\begin{cases} r = \left| d_t - d_{t0} \right|, d_{t0} < d_t \\ r = -\left| d_t - d_{t0} \right|, d_{t0} > d_t \end{cases} \tag{10}$$

where d_{t0} denotes the actual distance to the target, while d_t represents the theoretically expected distance.

$$d_{t0} = \sqrt{\left(t_x - p_{x0}\right)^2 + \left(t_y - p_{y0}\right)^2}$$
$$d_t = \sqrt{\left(t_x - p_x\right)^2 + \left(t_y - p_y\right)^2}$$

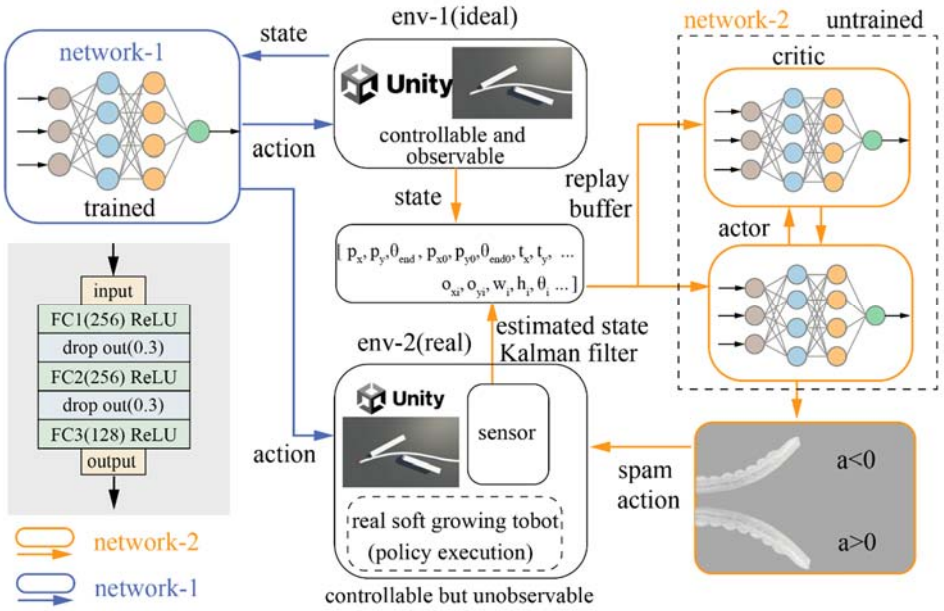

Figure S9. Perception and error correction network

The main framework of the algorithm is illustrated in Figure S9. It comprises a two-layer network structure. The first layer is the pre-trained pre-bending action selection network, as described in the previous section. The output of this network serves as the input to the second layer: the perception and error correction network, which adjusts PAM actuation to compensate for errors. The algorithm operates in two parallel environments: an ideal scenario and a simulated realistic scenario. The ideal

environment is both fully controllable and observable, whereas the simulated real-world environment is controllable but only partially observable. In the simulated realistic scenario, the robot's end coordinates, used as state variables, are estimated by fusing gyroscope-based end-attitude data and encoder-based length measurements via a Kalman filter. Additionally, noise is introduced into the output of the first network layer to simulate manufacturing errors in pre-bending.

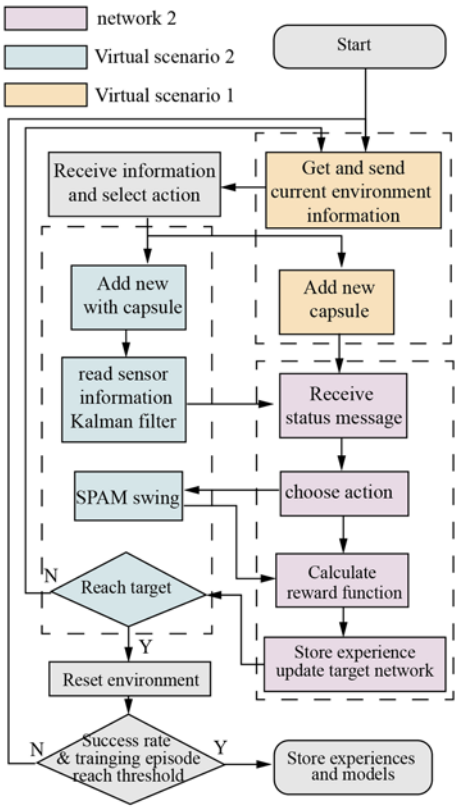

Figure S10. Flowchart of the environmental perception and error correction algorithm

The algorithm flowchart is illustrated in Figure S10. Initially, the trained Network-1 determines the pre-bending action of the soft growing robot based on environmental information extracted from the virtual scene. Following this decision, capsule bodies are added in both the virtual environment and the simulated realistic environment. In the simulated realistic environment, state variables are acquired using Kalman filtering with multi-sensor fusion. Subsequently, Network-2 processes these state variables from both environments and selects the appropriate PAM action. Based on this decision, the PAMs are activated in the simulated realistic environment to further minimize errors. Next, the reward function is computed based on the PAM movement results, and the target network is updated accordingly. The system then evaluates whether the target has been reached. If the target is achieved, the current scene is reset; otherwise, the process loops back to the environmental information retrieval step.

Following this, the success rate is assessed. If the predefined threshold or training episode is met, the training process concludes, and the current experience and model are stored. Otherwise, the next training iteration commences.

5 Ablation and Contrast Experiments

5.1 Ablation Experiment

The pre-bending action selection network of the soft growing robot incorporates both residual blocks and self-attention mechanisms to enhance the expressiveness and stability of policy learning. To evaluate the effectiveness of these components, we conducted ablation studies and systematically compared the results with simplified network architectures. As shown in Figure S11, the network integrating both residual blocks and attention mechanisms achieves the best performance, exhibiting the highest average reward and success rate, along with fast and stable convergence. The synergy between the two modules significantly improves the feature extraction and decision-making capabilities of both the policy and value networks.

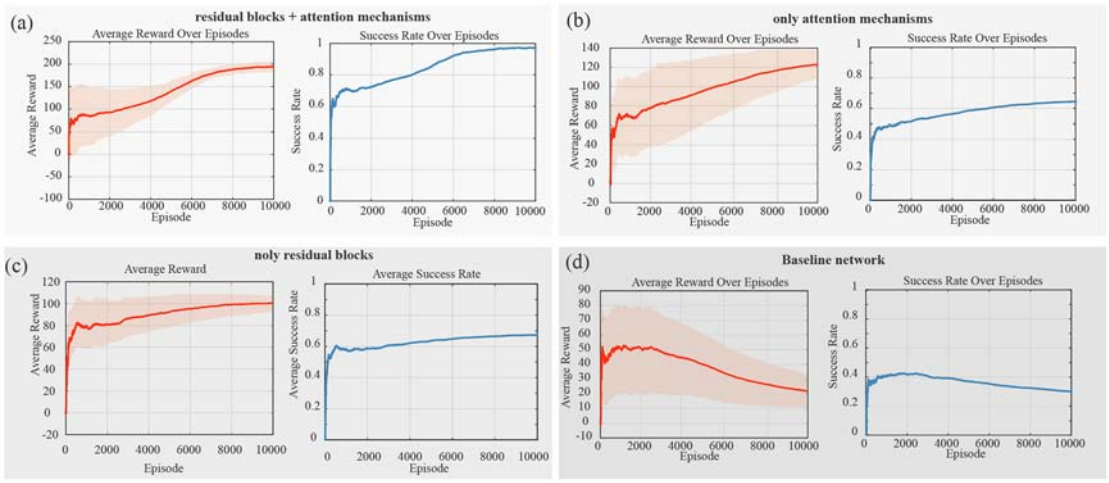

Figure S11. Ablation experiment results (a) Residual blocks combined with self-attention mechanism. (b) Self-attention mechanism only. (c) Residual blocks only. (d) Baseline network without residual blocks or attention mechanisms.

When only the attention mechanism is used, the reward and success rate still show a steady upward trend, but convergence is slower. This indicates that attention helps capture critical state features, but in the absence of residual connections, network stability is compromised, limiting further performance gains. In contrast, the network with only residual blocks quickly achieves high rewards in the early training phase but

plateaus thereafter, suggesting that residuals mainly improve training stability and mitigate gradient vanishing, with limited impact on long-term policy precision. The baseline network without any architectural enhancements performs the worst, showing no significant improvement in reward or success rate, and sometimes even a decline, indicating poor convergence in complex tasks. In summary, residual blocks and attention mechanisms each contribute uniquely to network performance, and their combination significantly enhances the efficiency and effectiveness of policy learning.

Table S1. Ablation experiment results(5 training results, standard deviation in parentheses)

	Final reward	Final success rate/（%）
Residual blocks + self-attention	194.94(5.31)	95.94(2.49)
Self-attention noly	122.93(7.12)	64.40(4.89)
Residual blocks only	100.58(7.36)	67.34(6.94)
Baseline network	21.93(14.57)	30.03(6.23)

5.2 Contrast Experiment

To evaluate the performance of the proposed dual-layer DDPG reinforcement learning framework, this section presents a comparative analysis of the pre-bending action selection network and the error correction network against two widely used algorithms: Soft Actor-Critic (SAC) and Proximal Policy Optimization (PPO). For fairness, all comparative experiments were conducted under consistent conditions, including identical network architectures, reward functions, training episodes, and learning rates.

For the pre-bending action selection network, the DDPG algorithm exhibited a faster convergence rate, with steadily increasing rewards and stabilizing policies, ultimately achieving a success rate of 95%. The SAC algorithm showed faster early-stage convergence and maintained stable performance in the later stages. While its overall performance was competitive, it was slightly inferior to DDPG, achieving an average success rate of around 79%. In contrast, PPO demonstrated stable training behavior throughout, but its convergence was significantly slower and long-term performance remained suboptimal, with a final success rate of approximately 78%.

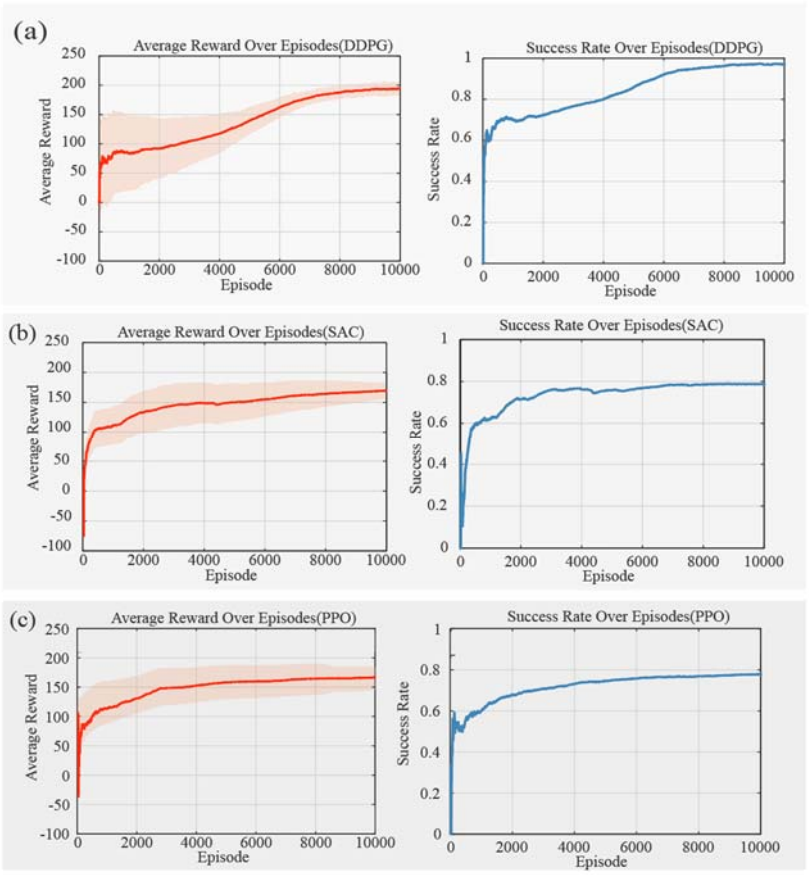

Figure S12. Comparative results of the pre-bending action selection network using different reinforcement learning algorithms. (a) DDPG (b) SAC (c) PPO

Table S2. Comparative results of the pre-bending action selection network(5 training results, , standard deviation in parentheses)

	Final reward	Final success rate(%)	Standard deviation
DDPG	194.94(5.31)	95(2.49)	11.5
SAC	169.32(9.84)	78.92(4.54)	12.6
PPO	167.02(9.23)	77.84(4.49)	17.6

For the perception and error correction network, the DDPG algorithm demonstrated superior overall performance. Although it exhibited considerable fluctuations in both average reward and success rate during the early exploration phase, it achieved strong convergence and a high average success rate (95%) in the later stages of training. In contrast, the SAC algorithm showed a rapid increase in rewards during the initial training phase. However, its exploration strategy tends to be more aggressive, and its convergence was comparatively slower. Nevertheless, SAC ultimately achieved a relatively high level of performance, with a success rate of approximately 80%. The PPO algorithm exhibited high training stability, with a steadily increasing average reward. However, due to its lower sample efficiency and slower convergence speed, its

overall performance lagged behind, achieving a success rate of only 66%. It required significantly more training iterations to reach convergence.

In summary, while all three algorithms exhibit unique strengths, DDPG offers the best overall performance and success rate for this task. It is particularly well-suited to continuous high-precision control scenarios with high-dimensional state spaces and low-dimensional action spaces.

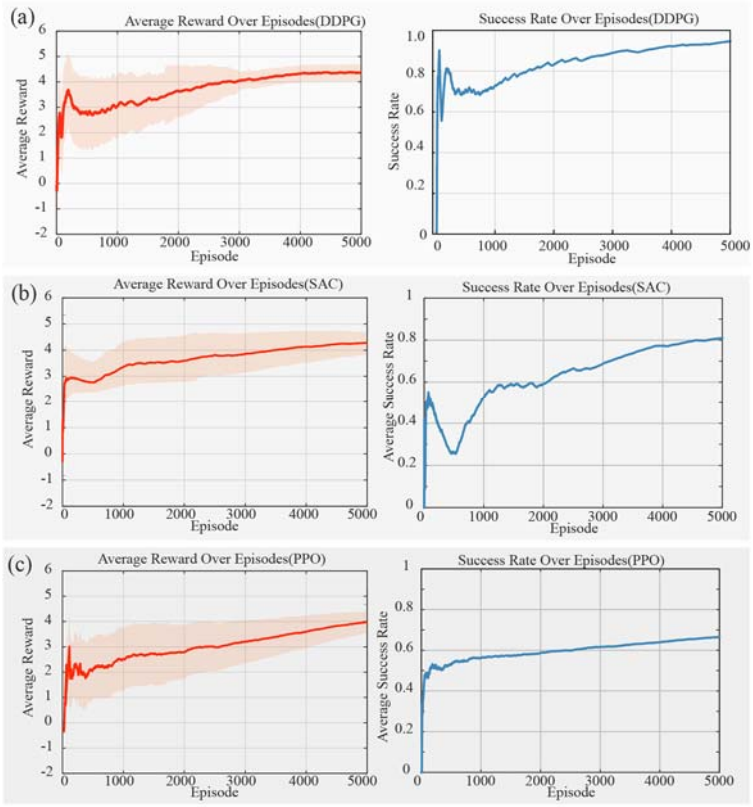

Figure S13. Comparative results of the error correction network across different reinforcement learning algorithms. (a) DDPG (b) SAC (c) PPO

Table S3. Comparative results of the error correction network(5 training results)

	Final reward	Final success rate/(%)	Standard deviation
DDPG	4.37(0.079)	94.64(2.58)	0.32
SAC	4.18(0.103)	80.08(3.68)	0.44
PPO	3.97(0.106)	66.57(4.31)	0.41

6 Experimental Verification

This section presents the complete motion process of the soft growing robot in confined environments, followed by a detailed analysis of the experimental results.

6.1 Experimental Verification of Pre-bending Action Selection Network

To further demonstrate the effectiveness of the pre-bending action selection network, it was tested in a real-world scenario. First, the required pre-bending actions for the soft growing robot were obtained within a virtual environment using the trained network.

Since the first-layer network is trained and validated in an idealized simulation, no modeling or sensor errors are present during training. However, in real-world implementations, pre-bending introduces inevitable fabrication errors, which typically fall within a certain tolerance range. To minimize the gap between theoretical assumptions and practical outcomes, we employed a manual selection process: after manufacturing each pre-bending segment, its actual bending angle was measured. If the deviation between the measured and target bending angles exceeded a predefined threshold, the segment was re-fabricated until the error fell within acceptable limits.

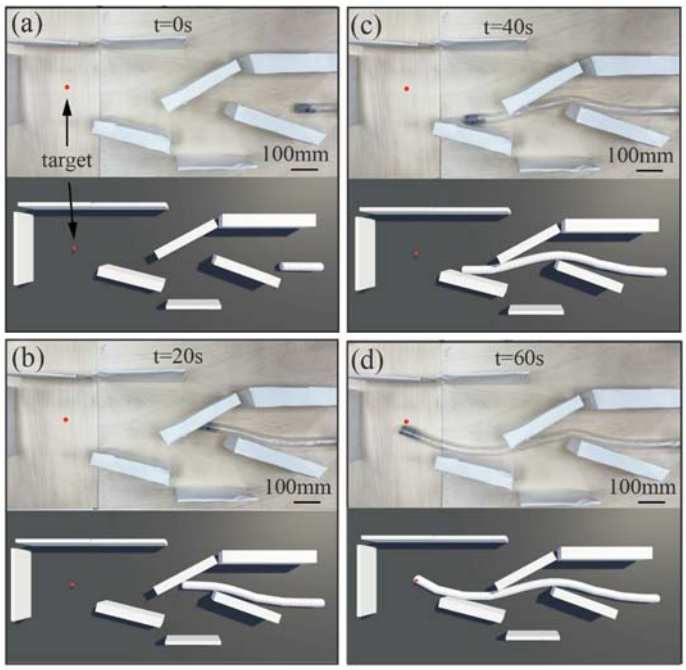

Figure S14. The soft growing robot navigates in a confined environment through pre-bending

As shown in Figure. S14, the soft growing robot navigates into a confined space by following pre-bending configurations determined by the action selection network. This interaction-based navigation strategy demonstrates the robot's inherent compliance and adaptability. The overall robot shape and its end-effector trajectory

show a high degree of consistency between simulation and real-world scenarios. During the robot's motion, we estimate the end-effector trajectory by combining vision-based tracking with multi-sensor fusion using a Kalman filter. As illustrated in Fig. S15, we present the deviation between theoretical and actual trajectories during a representative trial. Based on the results of 10 experiments, we compute the standard deviation of the path-following error. When the robot reaches a length of 1300 mm, the maximum trajectory deviation remains below 30 mm, and the standard deviation of the tracking error is 5.13 mm.

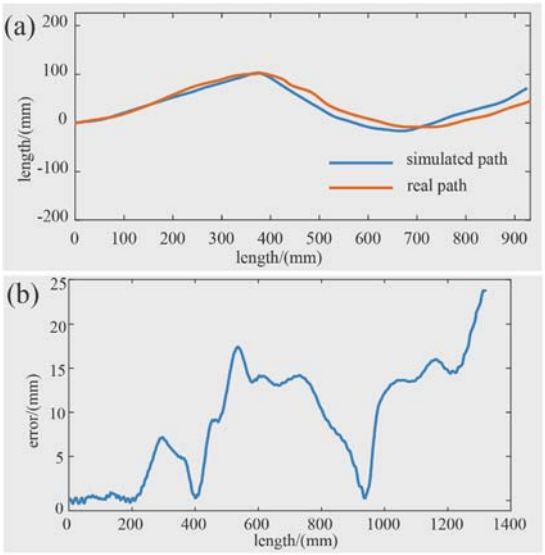

Figure S15. (a) Theoretical trajectory and actual trajectory (b) Error between theoretical and actual trajectories

6.2 Experimental Verification of Perception and Error Correction Network

To verify the effectiveness of the algorithm, we replace the simulated scenario in Figure S9 with a real-world scenario. First, in the simulated scenario, the pre-trained Network 1 determines actions, which are then applied to set the pre-bending of the soft growing robot. Notably, unlike the previous validation of the pre-bending network, explicit verification of the bending angle to minimize error is unnecessary. During movement, the trajectory of the soft growing robot is obtained using a sensor fusion algorithm that integrates gyroscope and encoder data.

This trajectory is then compared with the ideal trajectory from the Unity

simulation to determine the discrepancy between the ideal and actual trajectories. The computed error is input into the perception and error correction network to generate control signals for the PAMs. Additionally, in the simulated scenario, the pre-bending angles are set to match those in the real-world setup rather than being randomized based on Gaussian noise. The experimental validation of the perception and error correction network is shown in Figure 14, which presents the ideal, simulated, and real-world scenarios at key points.

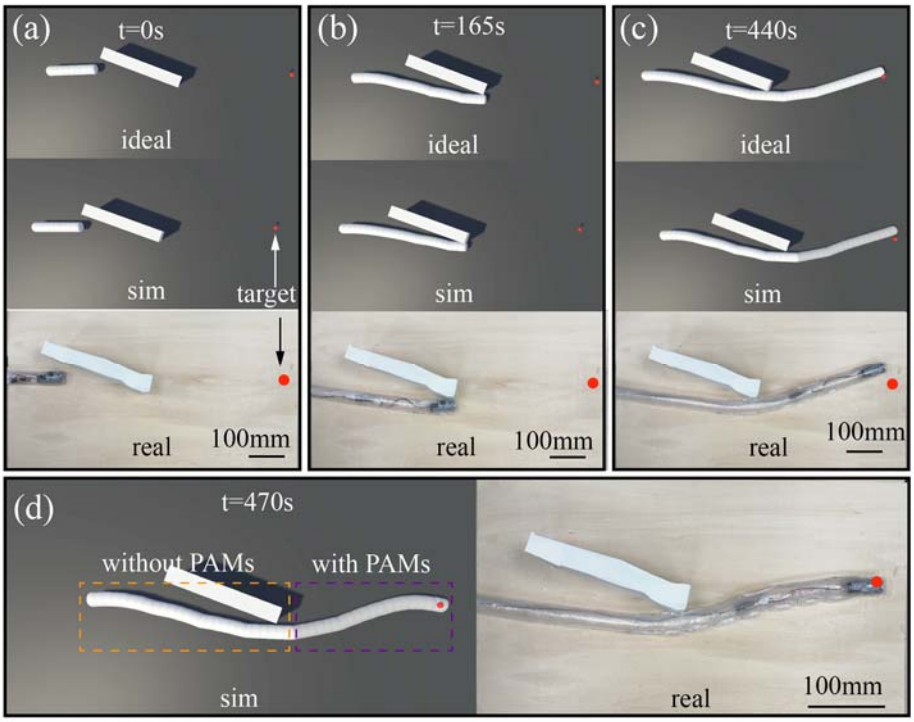

Figure S16: Experimental validation of the perception and error correction network: (a) Initial state (c) Approaching the target point with some error, (d) Error correction via pneumatic artificial muscle actuation.

As the error accumulates, when the robot in the ideal scenario moves to position C near the target, the end of the soft growing robot in the real-world scenario still deviates slightly from the target. At this point, the trained error correction network adjusts the movement of the PAMs to compensate for the error. The motion trajectory and path-following error of the soft growing robot are shown in Fig. S17. Before error correction, the maximum deviation between the theoretical and actual trajectories is less than 40 mm, with a standard deviation of 6.03 mm. After applying PAM actuation to correct the trajectory, the maximum deviation is reduced to less than 8 mm, and the standard deviation decreases to 1.47 mm. These results demonstrate that the proposed method can significantly reduce the trajectory error of pre-bent soft growing robots.

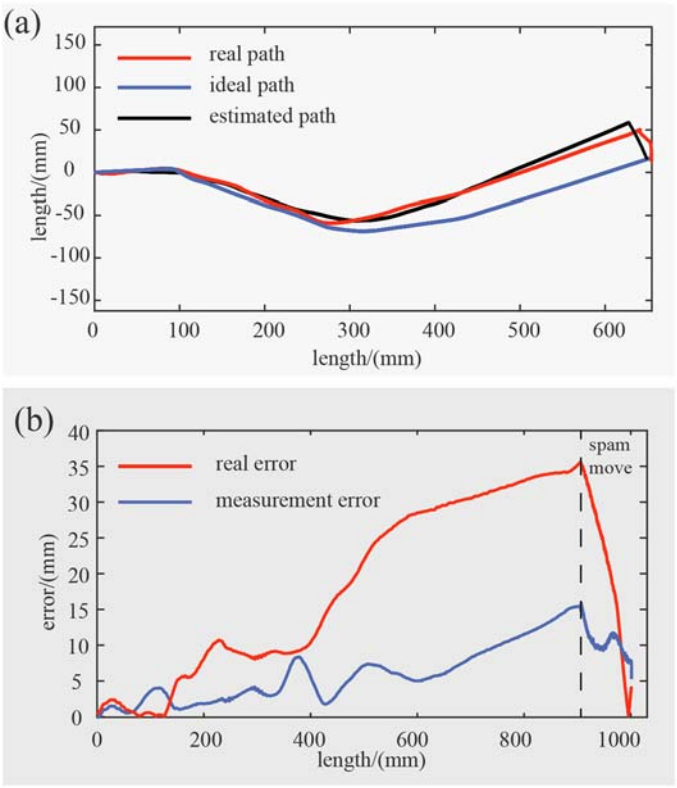

Figure S17 (a) Theoretical trajectory, actual trajectory, and estimated trajectory (b) Trajectory tracking error

References

[1] Greer J D, Blumenschein L H, Alterovitz R, et al. Robust navigation of a soft growing robot by exploiting contact with the environment. Int J Robot Res, 2020, 39: 1724–1738.

[2] Greer J D, Blumenschein L H, Okamura A M, et al. Obstacle-aided navigation of a soft growing robot. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia: IEEE, 2018, 4165–4172.

[3] Festo. Pneumatic cylinder ADN-25-30-I-P-A [EB/OL]. https://www.festo.com/us/en/a/8046299/, 2025-05-27.

[4] Coad M M, Blumenschein L H, Cutler S, et al. Vine robots. IEEE Robot Autom Mag, 2020, 27(3): 120–132.

[5] Greer J D, Morimoto T K, Okamura A M, et al. Series pneumatic artificial muscles (sPAMs) and application to a soft continuum robot. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, Singapore: IEEE, 2017, 5503–5510.

[6] Chen L, Gao Y, Wang S, et al. Physics-grounded differentiable simulation for soft growing

robots. arXiv preprint, arXiv:2501.17963, 2025.

[7] Jitosho R, Agharese N, Okamura A M, et al. A dynamics simulator for soft growing robots. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China: IEEE, 2021, 11775–11781.