Hierarchical RL-Guided Large-scale Navigation of a Snake Robot

Shuo Jiang, Adarsh Salagame, Alireza Ramezani, Lawson L.S. Wong

***Abstract*— Classical snake robot control leverages mimicking snake-like gaits tuned for specific environments. However, to operate adaptively in unstructured environments, gait generation must be dynamically scheduled. In this work, we present a four-layer hierarchical control scheme to enable the snake robot to navigate freely in large-scale environments. The proposed model decomposes navigation into global planning, local planning, gait generation, and gait tracking. Using reinforcement learning (RL) and a central pattern generator (CPG), our method learns to navigate in complex mazes within hours and can be directly deployed to arbitrary new environments in a zero-shot fashion. We use the high-fidelity model of Northeastern's slithering robot COBRA to test the effectiveness of the proposed hierarchical control approach.**

I. Introduction

Even on flat (zero elevation) ground, locomotion can be challenging in field applications due to uneven and unreliable terrain. Early research in snake robotics has focused on locomotion on flat ground and reproducing the unique locomotion of biological snakes. Typical gaits employed by snakes include lateral undulation, rectilinear motion, sidewinding, and concertina [1].

Lateral undulation is a means of movement along the longitudinal axis of the snake, relying on anisotropic friction to propel the snake forward as it moves in a sinusoidal pattern. This has been reproduced in [2], [3], with most works using wheels to provide anisotropic friction for body propulsion. Snakes use rectilinear motion in tight spaces by compressing and expanding the distance between scales to produce longitudinal locomotion [4], [5]. Sidewinding gait is employed by snakes to traverse on slippery and sandy slopes. It employs a sinusoidal gait that produces lateral motion. [6], [7] are notable early examples of this gait implemented. Finally, the Concertina gait is used in tight spaces where the range of motion is limited by coiling and uncoiling portions of the body to generate motion in the longitudinal axis [8]. In addition to these, non-snakelike gaits have been proposed to leverage the body's articulated nature to produce unique gaits such as the inchworm gait, slinky gait, lateral rolling gait, and tumbling locomotion [9], [10].

Research has shown that most gaits can be produced by having the snake robot's joints trace a series of sinusoids. Central pattern generators (CPG), as a dynamic oscillator, have been powerful in this regard, which can generate smoothly transitioning gaits controlled by varying the amplitude, frequency, and phases [11]–[15].

This work was supported by NSF Grant #2142519.

Northeastern University, Boston, MA 02115, USA {shuo.jiang, salagame.a, a.ramezani}@northeastern.edu ; lsw@ccs.neu.edu

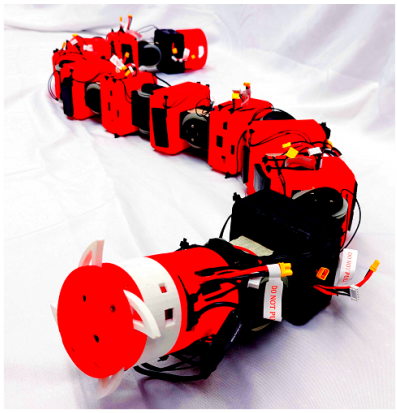


Fig. 1: Northeastern COBRA.

Using a fixed gait can only be applied to specific simple terrains. When the snake robot is deployed on unknown complex terrains, real-time gait adjustment based on environmental perception becomes the key to navigation. Model Predictive Control (MPC) has also been used to generate dynamic gaits that can avoid obstacles and minimize energy consumption [16]–[18]. However, an accurate dynamic model of the snake robot being in contact with the terrain is hard to obtain due to the characteristics of rich-contact. As a model-free control method, Reinforcement learning (RL) has received extensive attention in recent years and has been successfully applied to snake robots [19]–[22]. RL has the disadvantage that requires large amount of environment interaction, and directly applying RL methods to robots with a continuous action space with high degrees of freedom is inefficient. Therefore, the method of hierarchical RL has been proposed, and the combination of CPG and RL is one of them [19], [23]. RL is only responsible for the parameter adjustment of CPG, and CPG generates the gait that controls the robot joints. However, such hierarchical control scheme was only verified by simple tasks such as straight path tracking.

In this work, we aim to combat the challenge of locomotion control of high-DoF slithering robots by RL-CPG control scheme to enable the robot freely navigate in large-scale complex environments. We improved the previous RL-CPG method in the following aspects: first, adding a global planning module enables the robot to be deployed in new environments without retraining. Second, the robot uses ego-centric perception in training, which can be collected directly from on-body sensors without using external measuring tools, such as complex motion tracking systems. The latter is less applicable in outdoor environments. In addition, the global planning module decomposes complex

navigation problems into reusable simple local navigation problems, significantly reducing the problem's complexity and accelerating the training of RL.

II. COBRA's Hardware Overview

COBRA, shown in Fig. 2, is designed for space explorations [24]. It consists of eleven actuated joints. The front of the robot consists of a head module containing the onboard computing of the system, a radio antenna for communicating with a lunar orbiter, and an inertial measurement unit (IMU) for navigation. At the tail end, there is an interchangeable payload module containing a neutron spectrometer to detect water ice for our mission. The rest of the system consists of identical 1-DoF joint modules (Fig. 2) containing a joint actuator and a battery.

A. Head-tail Locking Mechanism

In addition to the eleven identical modules, COBRA features a distinct module at the snake's head, aptly referred to as the "head module," and, similarly, a "tail module" at the snake's tail end. The head module is shown in Fig. 1. The primary purpose of these unique modules is to connect to form a loop before the onset of a tumbling mode, a mode used for rapid mobility on lunar craters. The head module acts as the male connector and utilizes a latching mechanism to sit concentrically inside the female tail module.

The latching mechanism consists of a Dynamixel XC330 actuator, which sits within the head module and drives a central gear. This gear interfaces with the partially geared sections of four fin-shaped latching "fins." The curved outer face of each latching fin has an arc length equal to 1/4 of the circumference of the head module's circular cross-section. When the mechanism is retracted, these four fins form a thin cylinder that coincides with the head module's cylindrical face. A dome-shaped cap lies on the end of the head module so that the fins sit between it and the main body of the head module. Clevis pins are used to position the fins in this configuration. COBRA's tail module features a female cavity for the fins. When transitioning to tumbling mode, the head module is positioned concentrically inside the tail module using the joint's actuators, and the fins unfold into the cavity to lock the head module in place. For the head and tail modules to unlatch, the central gear rotates in the opposite direction, and the fins retract, allowing the system to return to sidewinding mode.

The choice for an active latching mechanism design stemmed from the design requirements and restrictions. Magnets were initially discussed as a passive latching option; however, they would not be effective in conjunction with the ferromagnetic regolith. Further, due to the need to stay in a latched configuration even when a large amount of force is applied to the system during tumbling, a passive system was not chosen, for there would be the risk of unlatching during tumbling.

III. Modeling

This section aims to establish a numerical framework to study the behavior of COBRA.

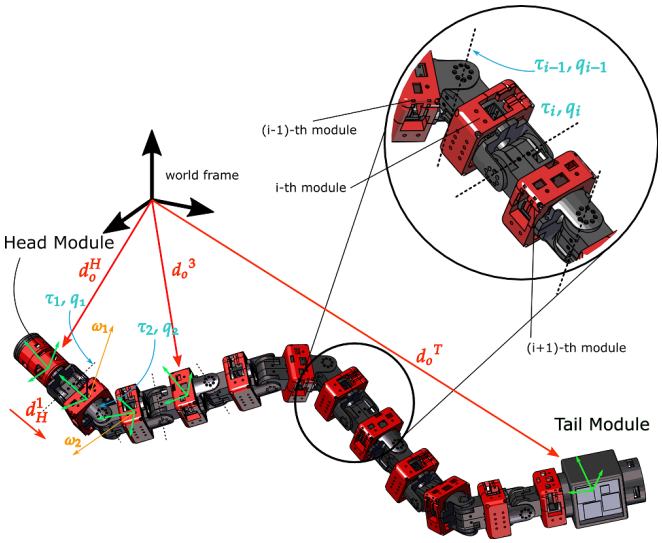


Fig. 2: Coordinate frames and parameters used for modeling COBRA.

A. Kinematics

Consider the configuration variable vector $q = [\dots, q_i, \dots, p_H^\top, q_H^\top]^\top$ which embodies the body angles q_i, head module position p_H, and orientations q_H. We use the Euler convention to find the rotation matrix R_H^0

$$R_H^0(q_H) = R_{z,q_z} R_{y,q_y} R_{x,q_x} \tag{1}$$

R_H^0 represents the points in the head frame with respect to the world frame. We consider the rotation matrices R_i^H

$$R_i^H = R_H^0 R_i^H, \ i = 1, \dots, N \tag{2}$$

R_i^H gives the expression of any points at each module with respect to the head frame. Using R_i^H and R_H^0, we obtain the forward kinematics equation and find the center of mass (CoM) position $p_{cm,i}$ of each module with respect to the world frame

$$p_{cm,i} = R_i^0 p_{cm,i}^i + d_i^0 \tag{3}$$

In this equation, d_i^0 denotes the world frame position of the body coordinates attached to i-th module. Angular velocity of the head module $\omega_H(t)$ and its relationship with the time derivative of the configuration variable $\dot{q}$ is given by

$$\hat{\omega}_H(t) = \dot{R}_H^0(t) R_H^0(t), \ \omega_H = \beta_H(q)\dot{q} \tag{4}$$

$\beta_H(q)$ is the head-module Jacobian matrix. The angular velocity of i-th module and its relation to $\dot{q}$ are given by

$$\hat{\omega}_i(t) = \dot{R}_i^0(t) R_i^0(t), \ \omega_i = \beta_i(q)\dot{q} \tag{5}$$

The world-frame velocity of i-th module CoM $v_{cm,i}$ can be obtained by

$$v_{cm,i}^0 = \hat{\omega}_i(t) R_i^0 p_{cm,i}^i + \dot{d}_i^0 = \sum_{i=1}^{N} \frac{\partial d_i^0}{\partial q_i} \dot{q}_i \tag{6}$$

B. Euler-Lagrange Formalism and Contact Model

The dynamical equations of motion of COBRA are given by

$$\begin{bmatrix} D_H & D_{Ha} \\ D_{aH} & D_a \end{bmatrix} \begin{bmatrix} \ddot{q}_H \\ \ddot{q}_a \end{bmatrix} + \begin{bmatrix} H_H \\ H_a \end{bmatrix} = \begin{bmatrix} 0 \\ B_a \end{bmatrix} u + \\ + \begin{bmatrix} J_H^\top \\ J_a^\top \end{bmatrix} F_{GRF} \tag{7}$$

where D_i, H_i, B_i, and J_i are partitioned model parameters corresponding to the head 'H' and actuated 'a' joints. F_{GRF} denotes the ground reaction forces. u embodies the joint actuation torques. The ground model used in our simulations is given by

$$\begin{aligned} F_{GRF} &= \begin{cases} 0 \quad \text{if } p_{C,z} > 0 \\ [F_{GRF,x},\, F_{GRF,y},\, F_{GRF,z}]^\top \quad \text{else} \end{cases} \\ F_{GRF,z} &= -k_1 p_{C,z} - k_2 \dot{p}_{C,z} \\ F_{GRF,i} &= -s_i F_{GRF,z}\, \mathrm{sgn}(\dot{p}_{C,i}) - \mu_v \dot{p}_{C,i} \quad \text{if} \quad i = x, y \\ s_i &= \left(\mu_c - (\mu_c - \mu_s) \mathrm{exp}\left(-|\dot{p}_{C,i}|^2 / v_s^2 \right) \right) \end{aligned} \tag{8}$$

where $p_{C,i}, \;\; i = x, y, z$ are the $x-y-z$ positions of the contact point; $F_{GRF,i}, \;\; i = x, y, z$ are the $x-y-z$ components of the ground reaction force assuming a point contact takes place between the robot and the ground substrate; k_1 and k_2 are the spring and damping coefficients of the compliant surface model; μ_c, μ_s, and μ_v are the Coulomb, static, and viscous friction coefficients; and, $v_s > 0$ is the Stribeck velocity.

While active posture control of COBRA in simulation and hardware is currently possible, we keep the scope of this work limited to simulations. The next steps involve projecting our control framework to the physical hardware, which will be reported in the sequel papers. Next, we outline our reinforcement-learning-based locomotion control of COBRA.

IV. Control

In this section, first, we present background on the Markov decision process (MDP) and central pattern generators (CPGs) before we outline the hierarchical control approach.

A. Markov Decision Process (MDP)

A MDP is a 4-tuple $M = \langle S, A, P, R \rangle$ where S is the set of states, A is the set of actions, $P\left(s_{t+1} | s_t, a_t\right)$ is the transition probability that action a in state s at time t that will lead to state s at time $t+1$, $R\left(a_t, s_t\right)$ is the distribution of reward when taking action a_t in state s_t. A policy $\pi\left(a_t | s_t\right)$ is defined as the probability distribution of choosing action a_t given state s_t. The learning goal is to find a policy π^* that maximizes the accumulated reward in given horizon T, $\pi^* = \underset{\pi}{\mathrm{argmax}} \underset{a_t, s_t \sim \pi}{\mathbb{E}} \left[\sum_{t=0}^{T-1} \gamma^t \cdot R\left(a_t, s_t\right) \right]$, where γ is discount factor. Reinforcement Learning (RL) algorithms are common choices to solve MDP problems.

B. Central Pattern Generators (CPGs)

CPG is a neural circuit in the vertebrate spinal cord that generates coordinated rhythmic output signals to control robot locomotion. CPG-based control methods have been successfully applied to many kinds of robots, such as multi-legged robots [23], [25] or snake robots [26]–[28]. Usually, to improve the terrain adaptability of CPG, optimization algorithms are often applied to adjust CPG parameters in real-time. As multiple CPG structures have been proposed, we adopted the structure in [27]. The dynamics of CPG are shown in Equation 9.

$$\begin{aligned} \dot{\varphi} &= \omega + \mathbf{A} \cdot \varphi + \mathbf{B} \cdot \theta \\ \ddot{r} &= a \cdot \left[\frac{a}{4} \left(R - r \right) - \dot{r} \right] \\ x &= r \cdot sin\left(\varphi \right) + \delta \end{aligned} \tag{9}$$

$$\mathbf{A} = \begin{bmatrix} -\mu_1 & \mu_1 & & & & \\ \mu_2 & -2\mu_2 & \mu_2 & & & \\ & & \ddots & & & \\ & & \mu_{n-1} & -2\mu_{n-1} & \mu_{n-1} & \\ & & & \mu_n & -\mu_n & \end{bmatrix} \tag{10}$$

$$\mathbf{B} = \begin{bmatrix} 1 & & & & & \\ -1 & 1 & & & & \\ & -1 & \ddots & & & \\ & & \ddots & 1 & & \\ & & & -1 & 1 & \\ & & & & -1 & \end{bmatrix} \tag{11}$$

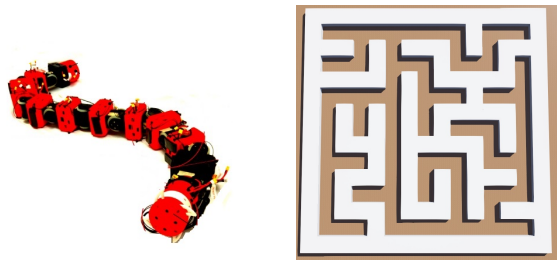
$\varphi \in \mathbb{R}^n$ and $r \in \mathbb{R}^n$ are internal states of CPG, n is the number of output channels, usually equals the number of robot joints. a and μ_i are hyperparameters that control the convergence rate. $R \in \mathbb{R}^n$, $\omega \in \mathbb{R}^n$, $\theta \in \mathbb{R}^{n-1}$, $\delta \in \mathbb{R}^n$ are inputs that control the desired amplitude, frequency, phase shift and offset. $x \in \mathbb{R}^n$ is the output sinusoidal waves of n channels.

C. Large-scale Navigation Scenario

In this work, we address the problem of COBRA navigation in large-scale scenarios. In the complex scene shown in Figure 3b, COBRA can move from any initial position to any goal position through locomotion. The controller is designed to directly adapt to new scenes in a zero-shot manner, given the scene map, without retraining. The COBRA robot in Fig. 3a has sensors that include joint encoders and an IMU in the head to measure linear accelerations. At the same time, we provide the relative pose between the robot and the target.

D. Four-layer Hierarchical Control Scheme

We designed a four-layer hierarchical control scheme to enable the snake robot with dynamics given by Eq. 7 to complete the above task. The controller structure is shown in Fig. 4. At the topmost level, global path planning uses tree search to find a path from the robot to the goal position and decomposes it into a series of waypoints. By adjusting



(a) (b)

Fig. 3: Large-scale navigation scenario considered in this paper. For a goal point in a maze-shaped world like this figure, COBRA performs path planning and tracking, including slithering in a straight path and making turns at junctions.

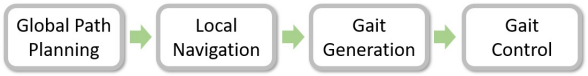


Fig. 4: Four-layer hierarchical control scheme.

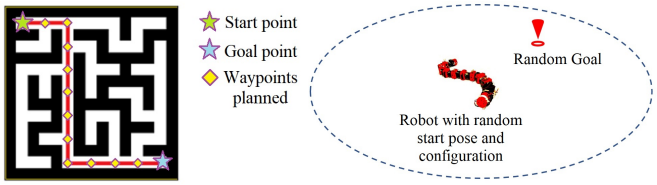
the CPG parameters, local navigation controls the robot's movement from one waypoint to another. Gait generation converts the CPG parameters into CPG output signals to provide the target positions of the robot joints. The lowest level of gait control uses motor feedback control to track the target gait signal.

1) Global Path Planning: Global path planning aims to find the shortest path in the domain, routing the robot to the destination. Given an occupancy grid of the domain, we use a tree search algorithm (A* algorithm particularly) to find the shortest path. We set several waypoints along the planned path with any two consecutive waypoints having a distance within a specific range (Fig. 5a). Then the problem turns out to be how to navigate the robot in a limited area to any goal (Fig. 5b). This setting dramatically reduces the complexity of the whole task compared with direct training to navigate the entire domain. The latter can be slow for training and less transferable. The local navigation task will be delivered to the lower-level RL controller.

2) Local Navigation: Due to the nature of snake-like locomotion (i.e., contact-rich problems), the description of its dynamic equation is extremely complicated; that is, the acyclic contact occurrence in Eq. 7 renders the control problem confounding. Most previous works adopted a simplified dynamics model and only assumed flat or regular ground. To improve the generalization of our method, we employed RL to solve the problem of local navigation proposed in the previous subsection and shown in Fig. 5b.

State space: the joint positions $\mathbb{R}^n$, IMU readings $\mathbb{R}^3$, spatial displacement between robot frame and waypoint frame $\mathbb{R}^3$, relative rotation between robot frame and waypoint frame are parameterized by axis-angle system $\mathbb{R}^4$, i.e., 21 DOFs in total. Notice the difference between our kinematics parameterization and previous works. We only use egocentric observations from the robot, where a complex motion capture system is not required as in [29], [30].

Action space: the action controls the parameter of two CPG components (see Gait Generation) of dimension $\mathbb{R}^7$. The action ranges can be summarized as (1) amplitude



(a) (b)

Fig. 5: For given start and goal points, the RL-guided locomotion control proposed in this work steers COBRA's eleven joints to reach the destination point.

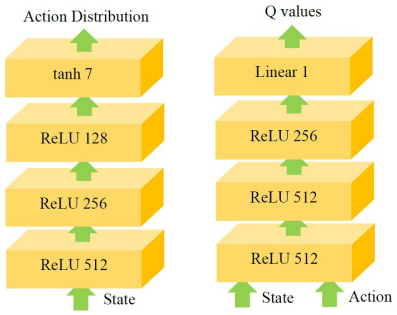


Fig. 6: Neural network structure of DDPG.

$R_1, R_2 \in [0, 1.5]$, (2) frequency $\omega_1 = \omega_2 \in [-0.1, 0.1]$, (3) phase shift $\theta_1, \theta_2 \in [-\pi, \pi]$, and (4) offset $\delta_1, \delta_2 \in [-0.1, 0.1]$.

Reward: We encourage the robot to reach the waypoint as soon as possible. The reward consists of the following terms:

$$\begin{aligned} r_1 &= \frac{1}{0.1 + d_t} \\ r_2 &= d_{t-1} - d_t \\ r_3 &= \|a_t - a_{t-1}\|_2 \end{aligned} \tag{12}$$

where d_t is the displacement between the robot frame and the waypoint frame. r_1 rewards nearer relative position and r_2 encourages approaching velocities. r_1 and r_2 work complementarily while $r_1 \to 0$ when robot is far away from the goal and $r_2 \to 0$ when robot is near the goal. r_3 ensures a smooth gait transition that penalizes the change of CPG parameters in consecutive planning steps.

Training Details: Since CPG produces a rhythmic gait, we need to reserve a certain amount of time for it to perform an effective gait, so RL should not change the CPG parameters too frequently. In this project, we let RL algorithm learn at a frequency of 0.5Hz, while the CPGs generate output signals at 50Hz to control the motors. Compared with the previous works where the learning frequency of RL is equal to the output frequency of CPG, or directly performing RL in the joint space, this design enables the RL algorithm to run more efficiently by reducing the rolling out length. Each episode lasts 160 seconds and a new goal will respawn in a random location in 8m×8m squared area. The total number of episodes is 40k that can be trained in a few hours. DDPG algorithm [31] is adopted as the backbone of RL, and the network structure is shown in Fig. 6.

3) Gait Generation: We use 2 CPGs to control both pitch and yaw-axis joints (Fig. 7). Each CPG has 4 parameters

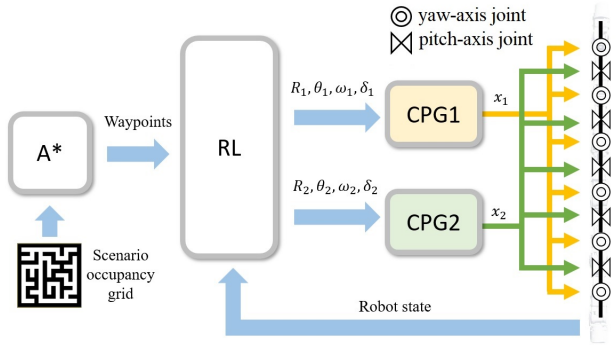


Fig. 7: RL block takes in the waypoints planned by the high-level global path planning component and the robot state, and it predicts the parameter of the two CPGs to guide the movement of the yaw-axis and pitch-axis joints.

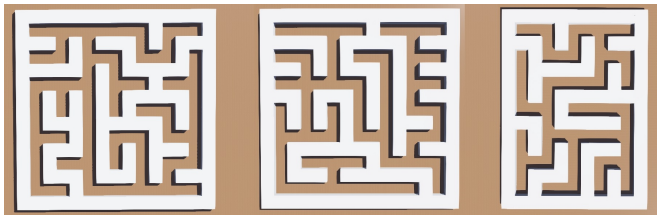


Fig. 8: Randomly generated maze layouts and sizes. The learned control scheme can be directly deployed without retraining.

to control its output sinusoidal waves, amplitude $R \in \mathbb{R}^k$, frequency $\omega \in \mathbb{R}^k$, phase shift $\theta \in \mathbb{R}^{k-1}$ and offset $\delta \in \mathbb{R}^k$. k is the number of channels a CPG governs. To generate a rhythmic movement pattern, we set each parameter of the four mentioned with one single value. Also, with prior knowledge, valid gait emerges only when the two CPGs have the same frequency.

4) Gait Control: For low-level motor control, we used PID controller for each joint to track the joint positions outputted by CPGs. The PID control generates the input torque u that steers the states in the model given by Eq. 7.

V. Simulation Results

Our control method can be directly deployed in a new domain in a zero-shot manner, unlike traditional RL methods that need to be retrained for different environments. To demonstrate this capability, we chose a randomized maze map generated using Kruskal's algorithm (Fig. 8). COBRA spawns at a random location on the map and autonomously navigates to a randomly generated destination. Two crucial modes of motion, straight motion (by sidewinding) and turning (by corner turning), are shown in Figs. 9 and 10. We also plot the trajectory of the CoM in Fig. 9 and 10. It can be seen that the turning motion is composed of multiple primitive motions, thus showing a more complex trajectory. The corresponding joint space movements are shown in Figs. 11a to 12b. We plot the pitch-axis and yaw-axis movements independently. Readers can see that the two sets of joints that are perpendicular and independent of each other adopt different motion patterns to enable the robot to move. Still, the motion patterns are all composed of sinusoidal curves.

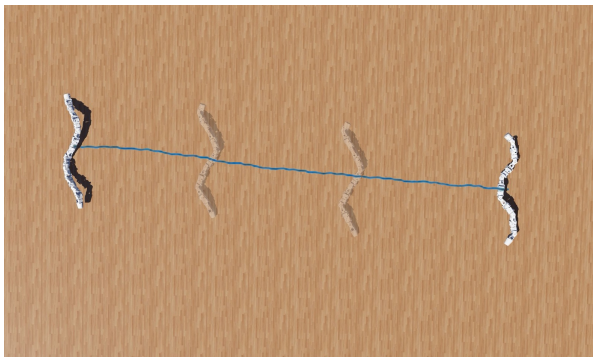


Fig. 9: CoM trajectory of sidewinding.

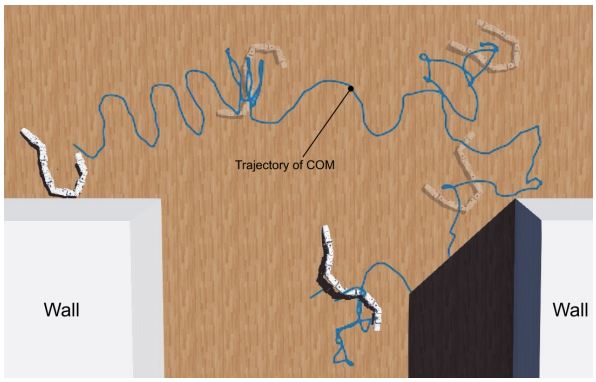
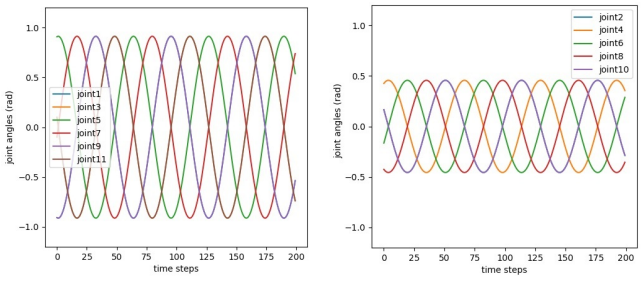
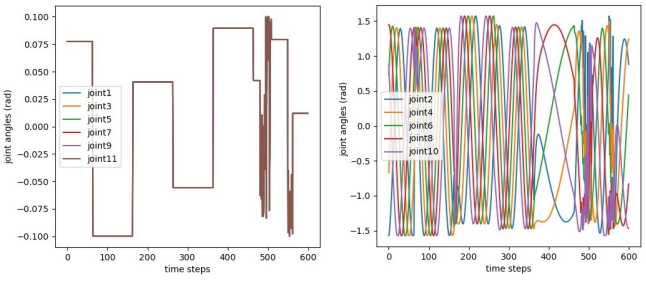


Fig. 10: CoM trajectory of corner turning.



(a) (b)

Fig. 11: Sidewinding: (a) pitch-axis joint trajectory; (b) yaw-axis joint trajectory.



(a) (b)

Fig. 12: Corner turning: (a) pitch-axis joint trajectory; (b) yaw-axis joint trajectory.

To show that the RL-CPG approach is an efficient scheme compared to applying RL directly in joint space (without CPG component), we compare the training time of both schemes on the same workstation (Fig. 13) with 3M running steps. We can see that the RL-CPG method significantly reduces the training time.

Fig. 13: Training time comparison.

VI. Conclusion

Classical joint space control design of slithering robots faces scalability challenges in large-scale navigation scenarios. In this paper, we design a four-layer motion control scheme for these robots. We tested the approach on the simulator of Northeastern's slithering robot, COBRA, to demonstrate its effectiveness. COBRA's model utilized the RL-CPG method for gait control and tree search for global navigation in complex maze-shaped environments. This design method not only dramatically shortens the training time of RL but also solves the problem of transferring the RL method to new environments.

References

[1] Y. Umetani and S. Hirose, "Biomechanical Study of Serpentine Locomotion," in *On Theory and Practice of Robots and Manipulators: Volume I*, ser. International Centre for Mechanical Sciences, Vienna: Springer, 1974, pp. 171–184.

[2] P. Wiriyacharoensunthorn and S. Laowattana, "Analysis and design of a multi-link mobile robot (Serpentine)," in *2002 IEEE International Conference on Industrial Technology, 2002. IEEE ICIT '02.*, vol. 2, Dec. 2002, 694–699 vol.2.

[3] S. Ma, N. Tadokoro, B. Li, and K. Inoue, "Analysis of creeping locomotion of a snake robot on a slope," in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 2, Sep. 2003, 2073–2078 vol.2.

[4] D. Rincon and J. Sotelo, "Ver-vite: Dynamic and experimental analysis for inchwormlike biomimetic robots," *IEEE Robotics & Automation Magazine*, vol. 10, no. 4, pp. 53–57, Dec. 2003.

[5] H. Ohno and S. Hirose, "Design of slim slime robot and its gait of locomotion," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, vol. 2, Oct. 2001, 707–715 vol.2.

[6] P. Liljebäck, Ø. Stavdahl, and K. Y. Pettersen, "MODULAR PNEUMATIC SNAKE ROBOT 3D MODELLING, IMPLEMENTATION AND CONTROL," *IFAC Proceedings Volumes*, 16th IFAC World Congress, vol. 38, no. 1, pp. 19–24, Jan. 2005.

[7] J. Burdick, J. Radford, and G. Chirikjian, "A 'sidewinding' locomotion gait for hyper-redundant robots," *Advanced Robotics*, vol. 9, no. 3, pp. 195–216, Jan. 1994.

[8] Y. Shan and Y. Koren, "Design and motion planning of a mechanical snake," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 4, pp. 1091–1100, Jul. 1993.

[9] A. Salagame, K. Gangaraju, H. K. Nallaguntla, E. Sihite, G. Schirner, and A. Ramezani, "Loco-manipulation with nonimpulsive contact-implicit planning in a slithering robot," *arXiv preprint arXiv:2404.08174*, 2024.

[10] A. Salagame, N. Bhattachan, A. Caetano, *et al.*, "How strong a kick should be to topple northeastern's tumbling robot?" *arXiv preprint arXiv:2311.14878*, 2023.

[11] S. Manzoor, Y. G. Cho, and Y. Choi, "Neural Oscillator Based CPG for Various Rhythmic Motions of Modular Snake Robot with Active Joints," *Journal of Intelligent & Robotic Systems*, vol. 94, no. 3, pp. 641–654, Jun. 2019.

[12] Z. Wang, Q. Gao, and H. Zhao, "CPG-Inspired Locomotion Control for a Snake Robot Basing on Nonlinear Oscillators," *Journal of Intelligent & Robotic Systems*, vol. 85, no. 2, pp. 209–227, Feb. 2017.

[13] Z. Bing, L. Cheng, G. Chen, F. Röhrbein, K. Huang, and A. Knoll, "Towards autonomous locomotion: CPG-based control of smooth 3D slithering gait transition of a snake-like robot," *Bioinspiration & Biomimetics*, vol. 12, no. 3, p. 035 001, Apr. 2017.

[14] E. Sihite, A. Kalantari, R. Nemovi, A. Ramezani, and M. Gharib, "Multi-Modal Mobility Morphobot (M4) with appendage repurposing for locomotion plasticity enhancement," *Nature Communications*, vol. 14, no. 1, p. 3323, Jun. 2023.

[15] A. Ramezani, P. Dangol, E. Sihite, A. Lessieur, and P. Kelly, "Generative Design of NU's Husky Carbon, A Morpho-Functional, Legged Robot," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 4040–4046.

[16] E. Hannigan, B. Song, G. Khandate, M. Haas-Heger, J. Yin, and M. Ciocarlie, "Automatic Snake Gait Generation Using Model Predictive Control," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 5101–5107.

[17] M. Nonhoff, P. N. Köhler, A. M. Kohl, K. Y. Pettersen, and F. Allgöwer, "Economic model predictive control for snake robot locomotion," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, Dec. 2019, pp. 8329–8334.

[18] H. Fukushima, T. Yanagiya, Y. Ota, M. Katsumoto, and F. Matsuno, "Model Predictive Path-Following Control of Snake Robots Using an Averaged Model," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 6, pp. 2444–2456, Nov. 2021.

[19] X. Liu, C. Onal, and J. Fu, "Reinforcement learning of a cpg-regulated locomotion controller for a soft snake robot," *arXiv preprint arXiv:2207.04899*, 2022.

[20] J. Shi, T. Dear, and S. D. Kelly, "Deep reinforcement learning for snake robot locomotion," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9688–9695, 2020.

[21] Y. Jia and S. Ma, "A coach-based bayesian reinforcement learning method for snake robot control," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2319–2326, 2021.

[22] Z. Bing, C. Lemke, L. Cheng, K. Huang, and A. Knoll, "Energy-efficient and damage-recovery slithering gait design for a snake-like robot based on reinforcement learning and inverse reinforcement learning," *Neural Networks*, vol. 129, pp. 323–333, 2020.

[23] G. Bellegarda and A. Ijspeert, "Cpg-rl: Learning central pattern generators for quadruped locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12 547–12 554, 2022.

[24] "Https://www.nasa.gov/feature/northeastern-university-slithers-to-the-top-with-big-idea-alternative-rover-concept,"

[25] N. D. Kent, D. Neiman, M. Travers, and T. M. Howard, "Improved performance of cpg parameter inference for path-following control of legged robots," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2022, pp. 11 963–11 970.

[26] X. Liu, R. Gasoto, Z. Jiang, C. Onal, and J. Fu, "Learning to locomote with artificial neural-network and cpg-based control in a soft snake robot," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 7758–7765.

[27] Z. Bing, L. Cheng, K. Huang, M. Zhou, and A. Knoll, "Cpg-based control of smooth transition for body shape and locomotion speed of a snake-like robot," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 4146–4153.

[28] X. Liu, C. Onal, and J. Fu, "Learning contact-aware cpg-based locomotion in a soft snake robot," *arXiv preprint arXiv:2105.04608*, 2021.

[29] P. Liljeback, K. Y. Pettersen, Ø. Stavdahl, and J. T. Gravdahl, "Experimental investigation of obstacle-aided locomotion with a snake robot," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 792–800, 2011.

[30] S. Hasanzadeh and A. A. Tootoonchi, "Ground adaptive and optimized locomotion of snake robot moving with a novel gait," *Autonomous Robots*, vol. 28, pp. 457–470, 2010.

[31] T. P. Lillicrap, J. J. Hunt, A. Pritzel, *et al.*, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[32] A. Salagame, N. Bhattachan, A. Caetano, *et al.*, *How Strong a Kick Should be to Topple Northeastern's Tumbling Robot?* arXiv:2311.14878 [cs, eess], Nov. 2023. DOI: 10.48550/arXiv.2311.14878. [Online]. Available: http://arxiv.org/abs/2311.14878 (visited on 12/02/2023).