Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Steering motion control of a snake robot via a biomimetic approach*

Wenjuan OUYANG[1], Wenyu LIANG[2], Chenzui LI[1], Hui ZHENG[3], Qinyuan REN[†‡1], Ping LI[1]

[1]*State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China*

[2]*Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117576, Singapore*

[3]*Zhejiang University of Science and Technology, Hangzhou 310023, China*

[†]E-mail: latepat@gmail.com

Received Sept. 11, 2018; Revision accepted Nov. 26, 2018; Crosschecked Jan. 8, 2019

Abstract: We propose a biomimetic approach for steering motion control of a snake robot. Inspired by a vertebrate biological motor system paradigm, a hierarchical control scheme is adopted. In the control scheme, an artificial central pattern generator (CPG) is employed to generate serpentine locomotion in the robot. This generator outputs the coordinated desired joint angle commands to each lower-level effector controller, while the locomotion can be controlled through CPG modulation by a higher-level motion controller. The motion controller consists of a cerebellar model articulation controller (CMAC) and a proportional-derivative (PD) controller. Because of the fast learning ability of the CMAC, the proposed motion controller can drive the robot to track the desired orientation and adapt to unexpected perturbations. The PD controller is employed to expedite the convergence speed of the motion controller. Finally, both numerical studies and experiments proved that the proposed approach can help the snake robot achieve good tracking performance and adaptability in a varying environment.

Key words: Snake robot; Central pattern generator; Cerebellar model articulation controller
https://doi.org/10.1631/FITEE.1800554 **CLC number:** TP242

1 Introduction

Over millions of years, snakes have developed excellent locomotion functions which have been shaped by natural selection. By bending their elongated and slender bodies in several special ways, snakes generate thrust from the interaction between the movements of their bodies and the environment, which is considerably superior to the mobility of conventional wheeled and tracked vehicles. Hence, inspired by the astonishing motor abilities of snakes, scientists have been developing new kinds of ground robots whose structures imitate the snake's morphology, namely, snake robots.

Most snake robots consist of serially connected modules that can bend in a manner similar to that of a snake. To drive these robots to be as agile as biological snakes, two control issues need to be addressed, locomotion generation and body motion control. Locomotion depends on producing coordinated movements of the connected modules of a snake robot, whereas body motion control focuses on controlling the whole body motion of the robot.

Previous studies of locomotion generation can be categorized into two methods from the perspective of cybernetics, a model-based approach and a central pattern generator (CPG) based approach. The model-based approach is to replicate the empirically observed body motion of crawling snakes with the movements of the multiple robotic modules connected by rotating joints. In this approach,

[‡] Corresponding author

[*] Project supported by the Zhejiang Provincial Natural Science Foundation of China (No. LQ16F030002)

ORCID: Qinyuan REN, https://orcid.org/0000-0001-9487-2675

the kinematic models are built to describe the body motion of snakes, and then the dynamics of the robotic movements are analyzed to obtain accurate mimicry of snake locomotion. Hirose (1993) presented serpentine locomotion of snakes via studies from the perspective of bionics. Ohno and Hirose (2001) experimentally verified that a 3D snake locomotion could be approximately described by three periodic functions. Based on the Lagrange method, Date et al. (2000) proposed a planar mathematical model for controlling a snake robot's locomotion autonomously. On the other hand, neurobiology studies pointed out that CPGs are neural networks, responsible for producing coordinated oscillatory signals to generate rhythmic movements of vertebrates such as swimming, flying, and walking (Marder and Bucher, 2001). Inspired by the advantages of stability and self-adaption of biological CPGs, artificial CPGs have been extensively investigated for locomotion generation of snake robots. Crespi and Ijspeert (2008) and Wu and Ma (2010) presented two CPG-based controllers that are constructed to generate serpentine locomotion for snake robots. A neural oscillator network based on CPGs was proposed for generating meandering locomotion in a snake robot (Inoue et al., 2004). Ryu et al. (2010) used a frequency-adaptive CPG model for undulatory snake-like locomotion. Although these approaches can be used to generate snake locomotion in snake robots, they cannot help the robots achieve a desired motion due to lack of feedback information.

To cause a snake robot to reach a given target, several feedback control approaches have been proposed. Mukherjee et al. (2017) addressed a velocity track problem of a snake robot via a sliding mode control (SMC) method. Liljeback et al. (2012) explored a cascaded-theory-based path-following controller that enables snake robots to track desired straight paths. Zhang et al. (2016) investigated a modified L_1 adaptive controller to control the direction of an underwater snake robot. Tanaka and Matsuno (2014) employed a model-based trajectory tracking controller for a head-raising snake robot using the features of kinematic redundancy.

The objective of this work is to address a steering motion control problem of a multi-link snake robot via a biomimetic approach. The proposed approach is inspired by the vertebrate biological motor system paradigm (Mattia et al., 2004). In this paradigm, the rhythmic motor pattern is generated at the spinal cord by a CPG network. A higher level of control is modulated by sensory feedback that can adapt the movement of the effectors (biological muscles) to unexpected perturbations through tuning the CPG network.

Inspired by this paradigm, a hierarchical control scheme, which includes an effector controller, a locomotion generator, and a higher-level body motion controller, is adopted. The effectors of the robot, the joint motors, are individually controlled by traditional proportional feedback controllers. In the locomotion generation part, an artificial CPG is implemented to generate serpentine locomotion for the snake robot with joint propulsion. This artificial CPG acts as a coupled oscillator with several tunable parameters. These tunable parameters are regulated by the higher-level body motion controller. The body motion controller consists of a cerebellar model articulation controller (CMAC) and a proportional-derivative (PD) controller. The CMAC, first proposed by Albus (1975), is a kind of neural network controller based on a model of the cerebellum. Due to the difficulty of developing an accurate analytical model that understands the robot crawls in dynamical environments, CMAC is proposed to compensate for the uncertainties and enhance the adaptability of the robot. Because of its fast learning ability, CMAC can approximate the perfect body motion control law. Meanwhile, the PD controller is employed to enhance the convergence speed of CMAC and guarantee the stability of the system at the early stage. Finally, the effectiveness and the tracking performance of the proposed approach are validated by numerical studies and experiments.

2 A multi-link snake robot

To mimic the movements of biological snakes, snake robots are usually designed in multi-link form.

A joint-link skeleton model of a snake robot that mimics the biological snake is shown in Fig. 1. In this skeleton model, the main body of the snake robot consists of multiple rigid links that are connected end to end. The joint generates torque to drive the relative motion between two connected links, playing a role similar to the muscle in biological snakes. The movements of the biological snake and the snake robot are generated via the interaction between their body motions and the environment.

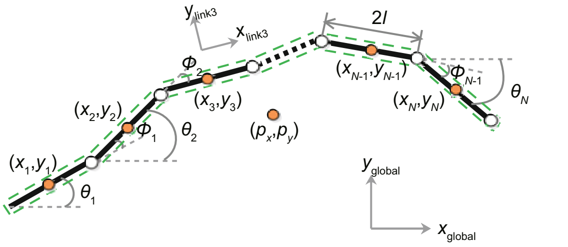

Fig. 1 The joint-link skeleton model of a snake robot

The notations shown in Fig. 1 are defined as follows:

Definition 1 (Link angle) The k^{th} ($k = 1, 2, \ldots, N$) link angle is the angle between the link and the global x axis with counterclockwise positive direction, denoted by $\theta_k \in \mathbb{R}$.

Definition 2 (Joint angle) The k^{th} ($k = 1, 2, \ldots, N-1$) joint angle is denoted by $\phi_k \in \mathbb{R}$ and defined as

$$\phi_k = \theta_k - \theta_{k+1}. \tag{1}$$

Definition 3 (Orientation) The orientation of the snake robot is defined as the average of the link angles and denoted by $\bar{\theta} \in \mathbb{R}$ as

$$\bar{\theta} = \frac{1}{N}\sum_{h=1}^{N}\theta_h. \tag{2}$$

All the link angles and all the joint angles are represented by vectors $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_N]^{\mathrm{T}} \in \mathbb{R}^N$ and $\boldsymbol{\phi} = [\phi_1, \phi_2, \ldots, \phi_{N-1}]^{\mathrm{T}} \in \mathbb{R}^{N-1}$, respectively. (p_x, p_y) and (x_k, y_k) are global coordinates of the center of mass (CM) of the k^{th} link and the robot, respectively. l represents the half length of a link.

According to the multi-link model, we develop a snake robot prototype (Fig. 2). The prototype is constructed using seven servo motors (Dynamics AX-12) and eight rigid plastic links, where the servo motors act like the joints to provide the required torque for the snake robot movements. The total length of the prototype is approximately 60 cm. At the first link, it is equipped with a micro controller board (AVR ATmega 32 microprocessor), a wireless Bluetooth module, and a Li-Po battery. The micro controller is used to exchange information through the wireless module, process sensor data, make movement decisions, and control all the servo motors. Moreover, each link of the snake robot body is attached to a pair of passive wheels in order to create the friction mechanism similar to a biological snake.

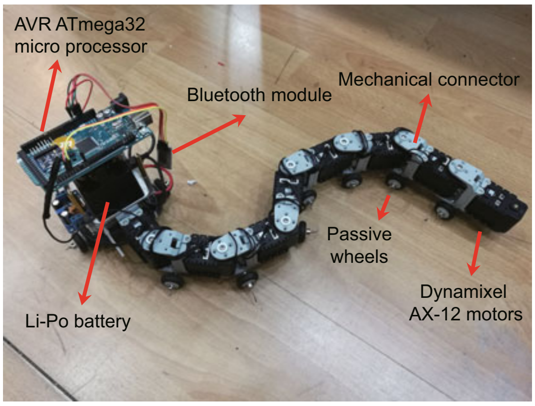

Fig. 2 The snake robot prototype

3 Steering motion control scheme

CPGs are primarily responsible for generating coordinated, rhythmic movements for locomotion of animals in real time. The patterns of these movements can be changed by the modulation in CPGs (Hooper, 2000). The modulation can be produced by a higher control level, such as a brainstem/cerebellum level, and the higher control level usually receives sensory feedback from the environment and accordingly modulates CPGs to enhance flexibility in response to unexpected perturbations. Inspired by this biological control mechanism, the steering motion control scheme for the snake robot prototype is shown in Fig. 3. It consists of three levels: a steering motion controller, a locomotion generator, and a joint controller. The steering motion controller is employed to control the orientation of the snake robot via the feedback error between the desired orientation and the feedback information. The steering motion controller will output the desired joint angle command of the first joint to achieve the desired steering motion to the locomotion generator. Then, the locomotion generator will

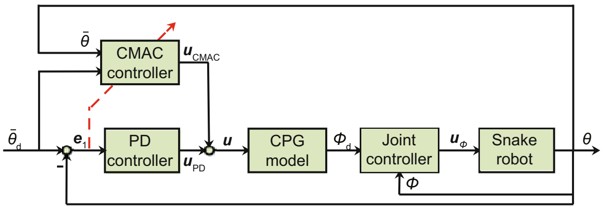

Fig. 3 Block diagram of the steering motion control scheme

generate the desired joint angle commands for all the joints and send these commands to the joint motors. Finally, the joint controller is used to control each joint angle to achieve the desired steering motion.

In the following subsections, the details of each level of the control scheme are presented from the inner level to the outer level.

3.1 Joint controller

In this snake robot, each joint angle is controlled by its corresponding joint motor. Significantly, the k^{th} joint angle ϕ_k is controlled by a joint controller embedded in the k^{th} servo motor. The joint control law is a position feedback controller that can be a proportional controller as shown below:

$$\boldsymbol{u}_{\phi} = k_{\phi}(\boldsymbol{\phi}_{\mathrm{d}} - \boldsymbol{\phi}), \tag{3}$$

where $\boldsymbol{\phi}_{\mathrm{d}}$ is the vector of the desired joint angles generated by the locomotion generator, $\boldsymbol{\phi}$ is the vector of the actual joint angles, k_{ϕ} is the control gain, and $\boldsymbol{u}_{\phi} \in \mathbb{R}^{N-1}$.

3.2 CPG-based locomotion generator

Various types of locomotion for snake robots have been introduced by Gray (1946), such as serpentine locomotion, sidewinding locomotion, concertina locomotion, and rectilinear locomotion. Among these types, serpentine locomotion is the fastest and the most common. In this locomotion, continuous waves are propagated backwards along the snake body from its head to its tail (Fig. 4). More details about the serpentine curve can be found in Hirose (1993).

Due to the similarity between the biological snake and the snake robot as well as the specific capability of the CPG-based model on coupling the dynamics of robots (Guo et al., 2018) and high stability under dynamic and changing conditions (Yu et al., 2014), a CPG model is adopted to generate serpentine locomotion in this study.

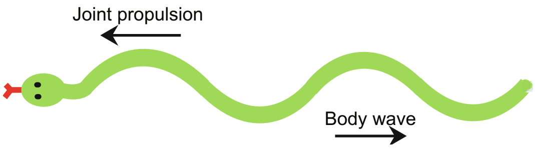

Fig. 4 Serpentine locomotion

3.2.1 CPG design

The CPG model used in the snake robot was designed based on the work presented by Crespi and Ijspeert (2008) (Fig. 5). The CPG model can be described as a neural network that consists of a double chain of multiple coupling Kuramoto oscillators (Crespi and Ijspeert, 2008). Each joint corresponds to an antagonistic oscillator pair. Therefore, the total number of oscillators is $2(N-1)$, where $N-1=7$ is the number of motorized joints in the snake robot. The right chain of the CPG is numbered from 1 to $N-1$, and the left chain of the CPG is numbered from N to $2(N-1)$. The CPG is implemented as the following coupled system:

$$\begin{cases} \dot{\beta}_i = \omega_i + \displaystyle\sum_{j \in T(i)} w_{ij} \sin(\beta_j - \beta_i + \varPsi_{ij}), \\ \ddot{r}_i = a_i \left(\dfrac{a_i}{4}(R_i - r_i) - \dot{r}_i \right), \\ x_i = r_i(1 + \cos\beta_i), \end{cases} \tag{4}$$

where $i = 1, 2, \ldots, 2(N-1)$ is the number of oscillators. β_i and r_i are the states of the phase and the amplitude in the i^{th} oscillator, respectively. ω_i determines the intrinsic frequency and $\omega_i = \omega$ for all oscillators. R_i is the amplitude, where the oscillators in the right chain of the CPG model are all equal to RR, while the oscillators in the left chain of the CPG model are all equal to RL. The coupling weight $w_{ij} = w_t$ for all oscillators and the phase bias $\varPsi_{ij}$ determine the phase lag among the oscillators. a_i is a positive constant and x_i is the output of the i^{th} oscillator.

According to the coupling relationship shown in Fig. 5, the i^{th} oscillator receives inbound couplings from the neighboring oscillators in the discrete set $T(i)$, for example, $T(1) = [2, 8]$, $T(5) = [4, 6, 12]$.

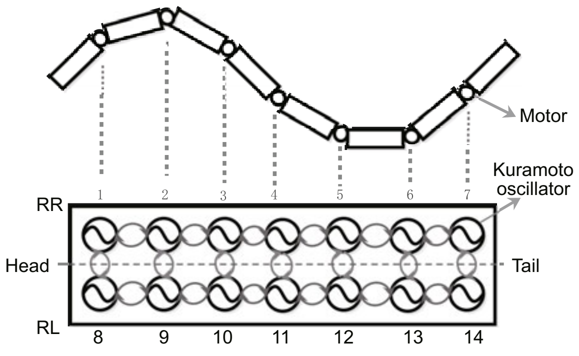

Fig. 5 The structure of the central pattern generator model used in the snake robot

The phase biases Ψ_{ij} are equal to π between the right chain and the left chain of the CPG, which is called antagonistic. In the same chain, the phase biases Ψ_{ij} between neighbor oscillators are set to ψ for the descending oscillators and $-\psi$ for the ascending oscillators, which is called coordination. The parameter is determined by $\psi = \frac{2\pi}{N-1}$.

The output of a CPG is expressed as

$$\phi_{\mathrm{d},k} = x_k - x_{k+N-1}, \tag{5}$$

where $\phi_{\mathrm{d},k}$ is the desired joint angle of the k^{th} joint, $k = 1, 2, \ldots, N-1$.

From Eqs. (4) and (5) with all the conditions mentioned above, the closed-form solution of $\phi_{\mathrm{d},k}$ is given by

$$\begin{aligned}\phi_{\mathrm{d},k}^{\infty} &= (\mathrm{RR}-\mathrm{RL}) + (\mathrm{RR}+\mathrm{RL})\cos(\omega t + k\psi + \psi_0)\\ &= \phi_0 + \bar{\phi}_k,\end{aligned} \tag{6}$$

where ψ_0 depends on the initial conditions of the system, (RL + RR) determines the oscillation amplitude, (RR − RL) affects the orientation of the snake robot, $\phi_0 = \mathrm{RR}-\mathrm{RL}$, and $\bar{\phi}_k = (\mathrm{RR}+\mathrm{RL})\cos(\omega t + k\psi + \psi_0)$, $k = 1, 2, \ldots, N-1$. $\bar{\boldsymbol{\phi}}_k = \left[\bar{\phi}_1, \bar{\phi}_2, \ldots, \bar{\phi}_{N-1}\right] \in \mathbb{R}^{N-1}$.

In the steering motion control scheme, the desired orientation of the snake robot can be set by applying different RR and RL values. Therefore, for steering motion control, the inputs of the CPG-based locomotion generator are shown below:

$$\begin{cases}\mathrm{RR} = \dfrac{\alpha+u}{2},\\[2ex] \mathrm{RL} = \dfrac{\alpha-u}{2},\end{cases} \tag{7}$$

where α is the designed amplitude of the joint angle oscillation, u is the output from the motion controller, and $\phi_0 = u$.

3.2.2 Locomotion generation simulations

To illustrate the CPG-based locomotion generator, numerical simulations are performed in MATLAB to study forward locomotion and steering locomotion. The simulation results are shown in Figs. 6 and 7.

In the forward locomotion simulation, the condition RR = RL is maintained. The explicit parameters of the amplitude, frequency, and coupling weight in the CPG model can be tuned to control the locomotion characteristics of the snake robot. Fig. 6 demonstrates that the CPG model can rapidly converge to the desired locomotion in a short transient time with different parameters. The ω and RR/RL influence the frequency and amplitude of the forward locomotion, respectively. When comparing the first curve and the second curve, it can be concluded that the coupling weight w_{ij} is involved in the transient process and that larger values lead to better performance.

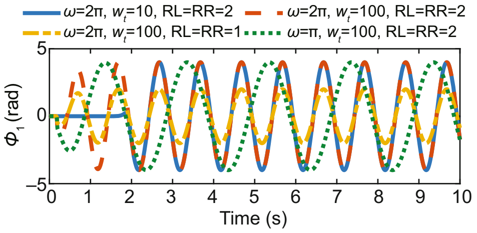

Fig. 6 CPG response curve with different parameters in forward locomotion

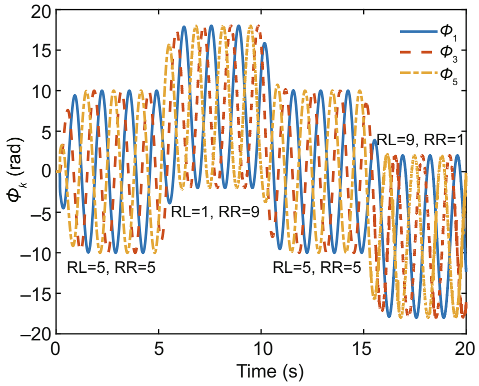

Fig. 7 CPG response curve with $\omega = 1.5\pi, w_t = 100$ in steering locomotion

In the steering locomotion simulation, we modify the values of RR and RL every five seconds. The joints k= 1, 3, and 5 are chosen for observation in this simulation. As shown in Fig. 7, the snake robot moves from forward locomotion to steering motion via translation of the joint angles along the y axis. Hence, it is verified that the steering motion can be controlled by setting different values of RR and RL to ensure that RR ≠ RL. In other words, the orientation of the snake robot is determined by the difference between RR and RL.

From the above simulation results, it can be observed that the CPG model can produce smooth steering trajectories even if the control parameters are abruptly changed. This property allows the system to run a motion controller and a locomotion generator in parallel, and the motion controller can regularly update the CPG parameters RR and RL online to achieve the desired steering motion.

3.2.3 Locomotion generation experiment

To validate the steering response, an experiment is conducted with the snake robot prototype described in Section 2. In the experiment, RR $= \alpha + u_\Delta$ and RL $= \alpha$ are set to the CPG model to allow the snake robot to achieve steering motion, where u_Δ is an adjustable input set manually in this experiment. Fig. 8 shows the relationship between the adjustable input and the orientation of the snake robot. As can be seen from the figure, different inputs correspond to different orientations that are almost linear. This characteristic can be beneficial to steering motion control.

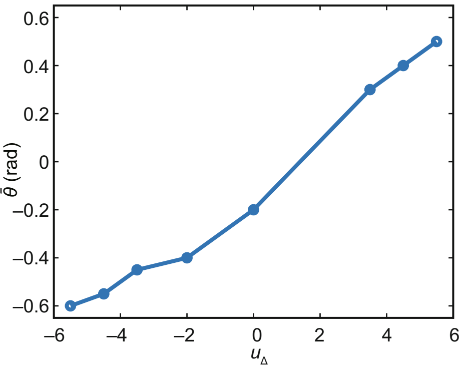

Fig. 8 Relationship between the adjustable input and the orientation of the snake robot

However, also note that a bias exists in this relationship, because environmental uncertainties (mainly the varying friction) affect the motion of the snake robot. Hence, it is difficult to achieve precise steering motion in such an open-loop control system (with only the CPG-based locomotion generator).

3.3 Motion controller

To solve the difficulty of achieving precise steering motion, one good solution is a motion controller to control the whole body motion of the snake robot with sensory feedback. Hence, a bioinspired steering motion controller is proposed, the details of which are given in the following.

3.3.1 Snake robot model

For the purposes of motion controller design and numerical study, a dynamic model of the snake robot is introduced. The planar snake robot model with $N + 2$ degrees-of-freedom (DOFs) is built based on the work done by Mukherjee et al. (2017), which is presented in detail in the following.

1. Kinematics. The CM of the snake robot in a global frame is given by

$$\boldsymbol{p} = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} \frac{1}{N}\sum_{k=1}^{N} x_k \\ \frac{1}{N}\sum_{k=1}^{N} y_k \end{bmatrix} = \frac{1}{N}\begin{bmatrix} \boldsymbol{e}^{\mathrm{T}}\boldsymbol{X} \\ \boldsymbol{e}^{\mathrm{T}}\boldsymbol{Y} \end{bmatrix}, \tag{8}$$

where $\boldsymbol{e} = [1, \ldots, 1]^{\mathrm{T}} \in \mathbb{R}^N$, $\boldsymbol{X} = [x_1, \ldots, x_N]^{\mathrm{T}} \in \mathbb{R}^N$ and $\boldsymbol{Y} = [y_1, \ldots, y_N]^{\mathrm{T}} \in \mathbb{R}^N$ denote the coordinates of the links.

The joint connections for all the links have to satisfy the two holonomic constraints shown below:

$$\begin{cases} \boldsymbol{DX} + l\boldsymbol{A}\cos\boldsymbol{\theta} = 0, \\ \boldsymbol{DY} + l\boldsymbol{A}\sin\boldsymbol{\theta} = 0, \end{cases} \tag{9}$$

with

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & & & \\ & \cdots & \cdots & & \\ & & \cdots & \cdots & \\ & & & 1 & 1 \end{bmatrix} \in \mathbb{R}^{(N-1)\times N},$$

$$\boldsymbol{D} = \begin{bmatrix} 1 & -1 & & & \\ & \cdots & \cdots & & \\ & & \cdots & \cdots & \\ & & & 1 & -1 \end{bmatrix} \in \mathbb{R}^{(N-1)\times N},$$

$$\sin\boldsymbol{\theta} = [\sin\theta_1, \ldots, \sin\theta_N]^{\mathrm{T}} \in \mathbb{R}^N,$$
$$\cos\boldsymbol{\theta} = [\cos\theta_1, \ldots, \cos\theta_N]^{\mathrm{T}} \in \mathbb{R}^N.$$

Hence, the linear velocities of the links can be derived according to Eqs. (8) and (9), given by

$$\begin{cases} \dot{\boldsymbol{X}} = l\boldsymbol{K}^{\mathrm{T}}\boldsymbol{S}_\theta\dot{\boldsymbol{\theta}} + \boldsymbol{e}\dot{p}_x, \\ \dot{\boldsymbol{Y}} = -l\boldsymbol{K}^{\mathrm{T}}\boldsymbol{C}_\theta\dot{\boldsymbol{\theta}} + \boldsymbol{e}\dot{p}_y, \end{cases} \tag{10}$$

where $\boldsymbol{S}_\theta = \mathrm{diag}(\sin\boldsymbol{\theta}) \in \mathbb{R}^{N\times N}$, $\boldsymbol{C}_\theta = \mathrm{diag}(\cos\boldsymbol{\theta}) \in \mathbb{R}^{N\times N}$, $\boldsymbol{K} = \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{DD}^{\mathrm{T}})^{-1}\boldsymbol{D} \in \mathbb{R}^{N\times N}$.

2. Friction. The planar snake robot generates forward propulsion via continuously swinging its body shape to create ground frictions. A ground friction model plays an important role in the dynamics of the snake robot. In this study, an anisotropic viscous friction model is employed, which means that

the tangential friction coefficient c_{t} is different from the normal friction coefficient c_{n}. The viscous friction force applied on all links is defined in the global frame in a vector form as shown below:

$$\begin{aligned}\boldsymbol{f}_{\mathrm{R}} &= \begin{bmatrix} \boldsymbol{f}_{\mathrm{R},x} \\ \boldsymbol{f}_{\mathrm{R},y} \end{bmatrix} \\ &= -\begin{bmatrix} c_{\mathrm{t}}(\boldsymbol{C}_\theta)^2 + c_{\mathrm{n}}(\boldsymbol{S}_\theta)^2 & (c_{\mathrm{t}} - c_{\mathrm{n}})\boldsymbol{S}_\theta\boldsymbol{C}_\theta \\ (c_{\mathrm{t}} - c_{\mathrm{n}})\boldsymbol{S}_\theta\boldsymbol{C}_\theta & c_{\mathrm{t}}(\boldsymbol{S}_\theta)^2 + c_{\mathrm{n}}(\boldsymbol{C}_\theta)^2 \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{X}} \\ \dot{\boldsymbol{Y}} \end{bmatrix},\end{aligned} \tag{11}$$

where $\dot{\boldsymbol{X}}$ and $\dot{\boldsymbol{Y}}$ are given in Eq. (10), and $\boldsymbol{f}_{\mathrm{R}} \in \mathbb{R}^{2N}$.

3. Dynamics. For a single link of the snake robot, the dynamic model of the k^{th} link is shown in Fig. 9, where $k = 1, 2, \ldots, N$. The k^{th} link is mainly subjected to the ground friction $\boldsymbol{f}_{\mathrm{R},k} \in \mathbb{R}^2$, the joint constraint forces $-h_{x,k-1}, -h_{y,k-1}, h_{x,k}$, and $h_{y,k}$, and the motor torques u_{k-1} and u_k. Significantly, the ground friction acts on the CM of the link, while the joint constraint forces compel the k^{th} link to connect to the $(k-1)^{\text{th}}$ and $(k+1)^{\text{th}}$ links. The motor torques are the driving force on the k^{th} link for generating angular motion.

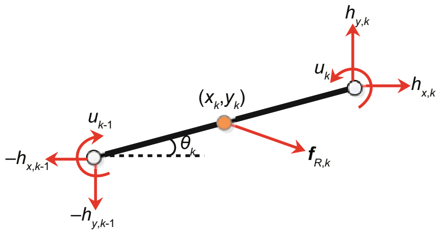

Fig. 9 Force analysis of the k^{th} link

Considering the two forces applied on the links, the force balance equations for the links can be given in the following vector forms:

$$\begin{cases} m\ddot{\boldsymbol{X}} = \boldsymbol{f}_{\mathrm{R},x} + \boldsymbol{D}^{\mathrm{T}}\boldsymbol{h}_x, \\ m\ddot{\boldsymbol{Y}} = \boldsymbol{f}_{\mathrm{R},y} + \boldsymbol{D}^{\mathrm{T}}\boldsymbol{h}_y, \end{cases} \tag{12}$$

where $\boldsymbol{h}_x = [h_{x,1}, \ldots, h_{x,N-1}]^{\mathrm{T}} \in \mathbb{R}^{N-1}$, $\boldsymbol{h}_y = [h_{y,1}, \ldots, h_{y,N-1}]^{\mathrm{T}} \in \mathbb{R}^{N-1}$, and m is the mass of each link.

The link accelerations can be derived by differentiating Eq. (9) twice with respect to time, given by

$$\begin{cases} \boldsymbol{D}\ddot{\boldsymbol{X}} = l\boldsymbol{A}(\boldsymbol{C}_\theta\dot{\boldsymbol{\theta}}^2 + \boldsymbol{S}_\theta\ddot{\boldsymbol{\theta}}), \\ \boldsymbol{D}\ddot{\boldsymbol{Y}} = l\boldsymbol{A}(\boldsymbol{S}_\theta\dot{\boldsymbol{\theta}}^2 - \boldsymbol{C}_\theta\ddot{\boldsymbol{\theta}}). \end{cases} \tag{13}$$

The acceleration of the CM can be obtained by differentiating Eq. (8) twice with respect to time, given by

$$\begin{bmatrix} \ddot{p}_x \\ \ddot{p}_y \end{bmatrix} = \frac{1}{N}\begin{bmatrix} \boldsymbol{e}^{\mathrm{T}}\ddot{\boldsymbol{X}} \\ \boldsymbol{e}^{\mathrm{T}}\ddot{\boldsymbol{Y}} \end{bmatrix}. \tag{14}$$

Substituting Eq. (12) into Eq. (14) and considering the fact that $\boldsymbol{e}^{\mathrm{T}}\boldsymbol{D}^{\mathrm{T}} = 0$, we have

$$\begin{bmatrix} \ddot{p}_x \\ \ddot{p}_y \end{bmatrix} = \frac{1}{N}\begin{bmatrix} \boldsymbol{e}^{\mathrm{T}}\ddot{\boldsymbol{X}} \\ \boldsymbol{e}^{\mathrm{T}}\ddot{\boldsymbol{Y}} \end{bmatrix} = \frac{1}{Nm}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{f}_{\mathrm{R}}, \tag{15}$$

where

$$\boldsymbol{E} = \begin{bmatrix} \boldsymbol{e} & \boldsymbol{0}_{N\times1} \\ \boldsymbol{0}_{N\times1} & \boldsymbol{e} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{2N\times2}.$$

Similarly, the torque balance equation for the links can be given in the following vector form:

$$J\ddot{\boldsymbol{\theta}} = \boldsymbol{D}^{\mathrm{T}}\boldsymbol{u}_\phi - l\boldsymbol{S}_\theta\boldsymbol{A}^{\mathrm{T}}\boldsymbol{h}_x + l\boldsymbol{C}_\theta\boldsymbol{A}^{\mathrm{T}}\boldsymbol{h}_y, \tag{16}$$

where J is the moment of inertia for each link and $\boldsymbol{u} = \boldsymbol{e}u \in \mathbb{R}^N$ is the vector form combining all the outputs from the motion controllers.

By solving Eq. (12) with Eq. (13), we can obtain $\boldsymbol{h}_x$ and $\boldsymbol{h}_y$. Finally, the complete dynamic model of the snake robot is included as shown in Eqs. (17) and (18):

$$\boldsymbol{M}_\theta\ddot{\boldsymbol{\theta}} = \boldsymbol{D}^{\mathrm{T}}\boldsymbol{u}_\phi - \boldsymbol{W}_\theta\dot{\boldsymbol{\theta}}^2 + \boldsymbol{R}_\theta\boldsymbol{f}_{\mathrm{R}}, \tag{17}$$

$$Nm\ddot{\boldsymbol{p}} = Nm\begin{bmatrix} \ddot{p}_x \\ \ddot{p}_y \end{bmatrix} = \begin{bmatrix} \boldsymbol{e}^{\mathrm{T}}\boldsymbol{f}_{\mathrm{R},x} \\ \boldsymbol{e}^{\mathrm{T}}\boldsymbol{f}_{\mathrm{R},y} \end{bmatrix} = \boldsymbol{E}^{\mathrm{T}}\boldsymbol{f}_{\mathrm{R}}, \tag{18}$$

with

$$\boldsymbol{M}_\theta = J\boldsymbol{I}_N + ml^2\boldsymbol{S}_\theta\boldsymbol{V}\boldsymbol{S}_\theta + ml^2\boldsymbol{C}_\theta\boldsymbol{V}\boldsymbol{C}_\theta \in \mathbb{R}^{N\times N},$$

$$\boldsymbol{W}_\theta = ml^2\boldsymbol{S}_\theta\boldsymbol{V}\boldsymbol{C}_\theta - ml^2\boldsymbol{C}_\theta\boldsymbol{V}\boldsymbol{S}_\theta \in \mathbb{R}^{N\times N},$$

$$\boldsymbol{R}_\theta = \begin{bmatrix} l\boldsymbol{S}_\theta\boldsymbol{K}, & -l\boldsymbol{C}_\theta\boldsymbol{K} \end{bmatrix} \in \mathbb{R}^{N\times2N},$$

$$\boldsymbol{V} = \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{D}\boldsymbol{D}^{\mathrm{T}})^{-1}\boldsymbol{A} \in \mathbb{R}^{N\times N}.$$

3.3.2 Motion controller design

Considering that $\phi_{k,\mathrm{d}} \approx \phi_{k,\mathrm{d}}^{\infty}$ (i.e., $\phi_d \approx \phi_d^{\infty}$), we can have the joint controller in the following form:

$$\boldsymbol{u}_\phi = k_\phi(\boldsymbol{\phi}_{\mathrm{d}} - \boldsymbol{\phi}) = k_\phi(\bar{\boldsymbol{\phi}} + \boldsymbol{\phi}_0 - \boldsymbol{D}\boldsymbol{\theta}), \tag{19}$$

where $\bar{\boldsymbol{\phi}} = [\bar{\phi}_1, \ldots, \bar{\phi}_{N-1}]^T \in \mathbb{R}^{N-1}$ and $\boldsymbol{\phi}_0 = [\phi_0, \ldots, \phi_0]^{\mathrm{T}} = \bar{\boldsymbol{e}}\phi_0 \in \mathbb{R}^{N-1}$ with $\bar{\boldsymbol{e}} = [1, \ldots, 1]^{\mathrm{T}} \in \mathbb{R}^{N-1}$.

Let $\boldsymbol{D}_\phi = \boldsymbol{D}^{\mathrm{T}}k_\phi$. Model (17) can be rewritten as

$$\boldsymbol{M}_\theta\ddot{\boldsymbol{\theta}} = \boldsymbol{D}_\phi(\bar{\boldsymbol{\phi}} - \boldsymbol{D}\boldsymbol{\theta}) + \boldsymbol{D}_\phi\phi_0 - \boldsymbol{W}_\theta\dot{\boldsymbol{\theta}}^2 + \boldsymbol{R}_\theta\boldsymbol{f}_{\mathrm{R}}. \tag{20}$$

Therefore, there exists a perfect controller $\boldsymbol{\phi}_0^*$ as shown below that can make the system asymptotically stable:

$$\begin{aligned}\phi_0^* = \boldsymbol{B}^{-1}[&-\boldsymbol{e}^{\mathrm{T}}\boldsymbol{M}_\theta^{-1}\boldsymbol{D}_\phi(\bar{\boldsymbol{\phi}}-\boldsymbol{D}\boldsymbol{\theta}) \\ &+\boldsymbol{e}^{\mathrm{T}}\boldsymbol{M}_\theta^{-1}\boldsymbol{W}_\theta\dot{\boldsymbol{\theta}}^2-\boldsymbol{e}^{\mathrm{T}}\boldsymbol{M}_\theta^{-1}\boldsymbol{R}_\theta\boldsymbol{f}_{\mathrm{R}} \\ &+N\ddot{\bar{\theta}}_{\mathrm{d}}+\lambda_{\mathrm{P}}(\bar{\theta}_{\mathrm{d}}-\bar{\theta})+\lambda_{\mathrm{D}}(\dot{\bar{\theta}}_{\mathrm{d}}-\dot{\bar{\theta}})],\end{aligned} \tag{21}$$

where $\boldsymbol{B}=\boldsymbol{e}^{\mathrm{T}}\boldsymbol{M}_\theta^{-1}\boldsymbol{D}_\phi\bar{\boldsymbol{e}}$ (significantly, it is defined that $\boldsymbol{B}^{-1}=\boldsymbol{0}$ while $\boldsymbol{B}=\boldsymbol{0}$) and $\bar{\theta}_{\mathrm{d}}$ is the desired orientation.

Applying the control law (21) on system model (20), we have

$$\boldsymbol{e}^{\mathrm{T}}\ddot{\boldsymbol{\theta}}=N\ddot{\bar{\theta}}_{\mathrm{d}}+\lambda_{\mathrm{P}}\tilde{\theta}+\lambda_{\mathrm{D}}\dot{\tilde{\theta}}=N\ddot{\bar{\theta}}, \tag{22}$$

where $\tilde{\theta}=\bar{\theta}_{\mathrm{d}}-\bar{\theta}$ is the orientation/steering error, $\dot{\tilde{\theta}}=\dot{\bar{\theta}}_{\mathrm{d}}-\dot{\bar{\theta}}$ is the error of the steering speed, and λ_{P} and λ_{D} are two designed positive parameters.

Let $\bar{k}_{\mathrm{D}}=\frac{\lambda_{\mathrm{D}}}{N}$, $\bar{k}_{\mathrm{P}}=\frac{\lambda_{\mathrm{P}}}{N}$. Then we obtain

$$\ddot{\tilde{\theta}}+\bar{k}_{\mathrm{D}}\dot{\tilde{\theta}}+\bar{k}_{\mathrm{P}}\tilde{\theta}=0, \tag{23}$$

where $\ddot{\tilde{\theta}}=\ddot{\bar{\theta}}_{\mathrm{d}}-\ddot{\bar{\theta}}$ and $\ddot{\bar{\theta}}=\frac{\boldsymbol{e}^{\mathrm{T}}\ddot{\boldsymbol{\theta}}}{N}$.

If λ_{P} and λ_{D} are designed properly, $s^2+\bar{k}_{\mathrm{D}}s+\bar{k}_{\mathrm{P}}$ will be Hurwitz stable. Then we can have

$$\lim_{t\to\infty}\tilde{\theta}=0.$$

This implies that the steering error can converge to zero and that the snake robot can successfully follow the desired steering motion.

In the snake robot, the actual input of the steering control is u, where $\boldsymbol{\phi}_0=\bar{\boldsymbol{e}}u$. Therefore, there also exists a perfect steering control law u^* that makes

$$\phi_0^*=u^*. \tag{24}$$

According to control laws (21) and (24), the perfect control law is a nonlinear function. Because the neural network can approximate the nonlinear function (Poggio and Girosi, 1990), a neural network controller can be used to approximate such a nonlinear control law in this study. Moreover, due to the model uncertainties and external disturbances (mainly because of the varying environments), an adaptive learning neural network controller can be a good solution. In this study, a cerebellar model articulation controller (CMAC) is employed as the neural network controller due to its simple structure and fast learning ability.

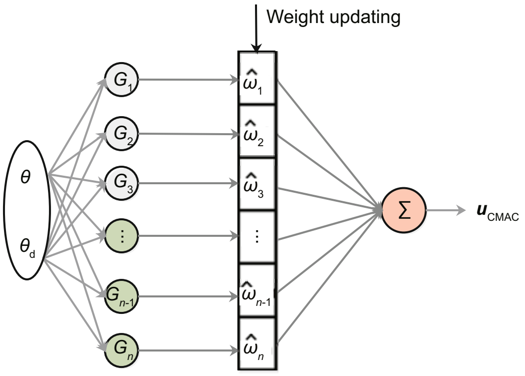

Fig. 10 The structure of the cerebellar model articulation controller

A conventional controller is applied along with this typical CMAC, which stabilizes the plant and helps the CMAC in its learning process in the early stage. To this end, a conventional proportional-derivative (PD) controller is employed with the CMAC.

Thus, a steering motion controller of the snake robot is designed as shown in Fig. 3. The steering motion controller is given by

$$u=u_{\mathrm{NN}}+u_{\mathrm{PD}}, \tag{25}$$

where u_{NN} is the CMAC-based neural network controller output and u_{PD} is the PD controller output.

In the following subsection, the neural network controller and PD controller are presented in detail.

1. Cerebellar model articulation controller

Inspired by the model of the cerebellum, CMAC was first proposed by Albus (1975). Experimental studies have proved that the cerebellum is involved in accurate motor learning and motor adaptation and cognition (Ito, 2000). In this study, the online learning characteristic of CMAC is used to achieve good steering motion control and enhance environmental adaptivity.

The CMAC network consists of three layers, namely, the input layer, intermediate layer, and output layer (Fig. 10). The input layer is responsible for importing the required information to the controller, which includes the desired orientation angle and the actual feedback information from the snake robot. The intermediate layer is used to locate the weights that need to be activated and update the weights to minimize the error. The output layer is used to calculate the sum of each neuron and output the calculated control signal to the system.

Significantly, the control output of CMAC is the linear combination of the activated weights in the weight memory $\hat{\boldsymbol{\omega}}$. The CMAC control law is given by

$$u_{\mathrm{CMAC}} = \sum_{p=1}^{n} \hat{\omega}_p G_p = \hat{\boldsymbol{\omega}}^{\mathrm{T}} \boldsymbol{\varphi}, \tag{26}$$

where $\hat{\boldsymbol{\omega}} = [\hat{\omega}_1, \hat{\omega}_2, \ldots, \hat{\omega}_n]^{\mathrm{T}} \in \mathbb{R}^n$ is the estimated weight vector and $\hat{\omega}_p$ denotes the weight stored in the p^{th} region of the weight memory. $\boldsymbol{\varphi}$ is the multidimensional basis function used for locating the weight under a multidimensional circumstance, designed as

$$\boldsymbol{\varphi}(s) = [G_1, G_2, \ldots, G_n]^{\mathrm{T}}, \tag{27}$$

with

$$\begin{cases} \varphi_{rl_r}(s_r) = \exp\left[-\dfrac{(s_r - \mu_{rl_r})^2}{\sigma_{rl_r}^2}\right], \\ G_l(s_1, s_2) = \displaystyle\prod_{r=1}^{2} \varphi_{rl_r}(s_r) = \varphi_{1l_1}(s_1)\varphi_{2l_2}(s_2), \\ l = 1, 2, \ldots, n \text{ and } r = 1, 2, \end{cases}$$

where the basis function is used to determine the activated regions for the corresponding input vector $s = [s_1, s_2]^{\mathrm{T}} = [\theta, \theta_{\mathrm{d}}]^{\mathrm{T}}$. $\varphi_{kl}(s_k)$ is the quantization distance of the k^{th} input s_k to the l^{th} region. μ_{kl} and σ_{kl} denote the center coordinate and the width of the l^{th} region, respectively. Significantly, μ_{kl} and σ_{kl} are designed based on the standard quantization interval shown below:

$$q_r = \frac{s_{r,\max} - s_{r,\min}}{R}, \tag{28}$$

where $r = 1, 2$. $s_{r,\max} = \max(s_r)$ and $s_{r,\min} = \min(s_r)$ denote the maximum and minimum of s_r, respectively. $R \in \mathbb{N}^+$ determines the number of quantized regions, $n = R + 1$ is the number of the quantized points, and q_r determines the quantization resolution.

When the activation of the region runs, the corresponding weights will be activated and the system will output the sum of the activated weights. In each learning cycle, the weights are adjusted by the following update law:

$$\dot{\hat{\boldsymbol{\omega}}} = \rho \boldsymbol{e}_\theta^{\mathrm{T}} \boldsymbol{P}_\theta \boldsymbol{B}_\theta \boldsymbol{\varphi}, \tag{29}$$

where $\rho > 0$ is a constant learning rate and $\boldsymbol{e}_\theta = [\tilde{\theta}, \dot{\tilde{\theta}}]^{\mathrm{T}}$ are the error state. $\boldsymbol{B}_\theta = \left[0, \ \frac{B}{N}\right]^{\mathrm{T}}$ and P_θ is the solution of $\boldsymbol{A}_\theta^{\mathrm{T}} \boldsymbol{P}_\theta + \boldsymbol{P}_\theta \boldsymbol{A}_\theta = -\boldsymbol{Q}_\theta \leq 0$ with $\boldsymbol{A}_\theta = \begin{bmatrix} 0 & 1 \\ -\bar{k}_{\mathrm{P}} & -\bar{k}_{\mathrm{D}} \end{bmatrix}$ and $\boldsymbol{Q}_\theta \geq 0$.

To further compensate for the approximation error ε between the joint control law (21) and the CMAC output (26), a compensator is proposed using a slide mode technique:

$$u_s = \hat{\varepsilon}_m \operatorname{sign}(\boldsymbol{e}_\theta^{\mathrm{T}} \boldsymbol{P}_\theta \boldsymbol{B}_\theta), \tag{30}$$

and its update law is

$$\dot{\hat{\varepsilon}}_m = \operatorname{Proj}(\hat{\varepsilon}_m, \rho_s |\boldsymbol{e}_\theta^{\mathrm{T}} \boldsymbol{P}_\theta \boldsymbol{B}_\theta|), \tag{31}$$

where $\operatorname{Proj}(\cdot)$ is the projection algorithm and $\operatorname{Proj}(\hat{\varepsilon}_m, \rho_s |\boldsymbol{e}_\theta^{\mathrm{T}} \boldsymbol{P}_\theta \boldsymbol{B}_\theta|) \geq \rho_s |\boldsymbol{e}_\theta^{\mathrm{T}} \boldsymbol{P}_\theta \boldsymbol{B}_\theta|$.

Assuming that there exists an optimal weight vector (29) that can minimize the approximation error ε, we have

$$\begin{cases} u^* - \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{\varphi} = \varepsilon, \\ u^* - \hat{\boldsymbol{\omega}}^{\mathrm{T}} \boldsymbol{\varphi} = \varepsilon + \tilde{\boldsymbol{\omega}}^{\mathrm{T}} \boldsymbol{\varphi}, \end{cases}$$

where $\tilde{\boldsymbol{\omega}} = \boldsymbol{\omega} - \hat{\boldsymbol{\omega}}$ is the difference between the optimal weight vector and the estimated weight vector. ε is the approximation error between the perfect nonlinear control law and the neural network controller, and is assumed to be bounded, i.e., $|\varepsilon| \leq \varepsilon_m$.

Substituting $u_{\mathrm{NN}} = u_{\mathrm{CMAC}} + u_s$ in system (22), we have

$$\dot{\boldsymbol{e}}_\theta = \begin{bmatrix} \dot{\tilde{\theta}} \\ \ddot{\tilde{\theta}} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -\bar{k}_{\mathrm{P}} & -\bar{k}_{\mathrm{D}} \end{bmatrix}}_{\boldsymbol{A}_\theta} \underbrace{\begin{bmatrix} \tilde{\theta} \\ \dot{\tilde{\theta}} \end{bmatrix}}_{\boldsymbol{e}_\theta} + \underbrace{\begin{bmatrix} 0 \\ \frac{B}{N} \end{bmatrix}}_{\boldsymbol{B}_\theta} (u_\phi^* - u_{\mathrm{NN}}). \tag{32}$$

Hence, the whole control system can be described as

$$\begin{aligned} \dot{\boldsymbol{e}}_\theta &= \boldsymbol{A}_\theta \boldsymbol{e}_\theta + \boldsymbol{B}_\theta (u^* - \hat{\boldsymbol{\omega}}^{\mathrm{T}} \boldsymbol{\varphi} - u_s) \\ &= \boldsymbol{A}_\theta \boldsymbol{e}_\theta + \boldsymbol{B}_\theta (\varepsilon + \tilde{\boldsymbol{\omega}}^{\mathrm{T}} \boldsymbol{\varphi} - u_s). \end{aligned} \tag{33}$$

We define the Lyapunov candidate function as

$$V = \frac{1}{2} \boldsymbol{e}_\theta^{\mathrm{T}} \boldsymbol{P}_\theta \boldsymbol{e}_\theta + \frac{1}{2\rho} \tilde{\boldsymbol{\omega}}^{\mathrm{T}} \tilde{\boldsymbol{\omega}} + \frac{1}{2\rho_s} \tilde{\varepsilon}_m^2, \tag{34}$$

where $\tilde{\varepsilon}_m = \varepsilon_m - \hat{\varepsilon}_m$.

Take the time derivative of Eq. (34):

$$\begin{aligned}\dot V &= \frac{1}{2}\boldsymbol{e}_\theta^{\mathrm{T}}(\boldsymbol{A}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta + \boldsymbol{P}_\theta\boldsymbol{A}_\theta)\boldsymbol{e}_\theta \\ &\quad + \boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta\boldsymbol{B}_\theta(\varepsilon + \tilde{\boldsymbol{\omega}}^{\mathrm{T}}\boldsymbol{\varphi} - u_s) \\ &\quad - \frac{1}{\rho}\tilde{\boldsymbol{\omega}}^{\mathrm{T}}\dot{\hat{\boldsymbol{\omega}}} - \frac{1}{\rho_s}\tilde{\varepsilon}_m\dot{\hat{\varepsilon}}_m \\ &\le -\frac{1}{2}\boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{Q}_\theta\boldsymbol{e}_\theta + \boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta\boldsymbol{B}_\theta(\varepsilon + \tilde{\boldsymbol{\omega}}^{\mathrm{T}}\boldsymbol{\varphi}) \\ &\quad - \hat{\varepsilon}_m|\boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta\boldsymbol{B}_\theta| - \boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta\boldsymbol{B}_\theta\tilde{\boldsymbol{\omega}}^{\mathrm{T}}\boldsymbol{\varphi} \\ &\quad - \tilde{\varepsilon}_m|\boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta\boldsymbol{B}_\theta| \\ &= -\frac{1}{2}\boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{Q}_\theta\boldsymbol{e}_\theta + \boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta\boldsymbol{B}_\theta\varepsilon - \varepsilon_m|\tilde{\boldsymbol{e}}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta\boldsymbol{B}_\theta| \\ &\le -\frac{1}{2}\boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{Q}_\theta\boldsymbol{e}_\theta + |\boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta\mathbf{B}_\theta|(|\varepsilon| - \varepsilon_m).\end{aligned} \tag{35}$$

As the inequality $|\varepsilon| \le \varepsilon_m$ holds in Eq. (32), it can be obtained that

$$\dot V \le -\frac{1}{2}\boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{Q}_\theta\boldsymbol{e}_\theta \le 0. \tag{36}$$

It is obvious that $\dot V$ is negative, which implies that $\boldsymbol{e}_\theta, \tilde{\boldsymbol{\omega}}, \tilde{\varepsilon}_m$ are bounded and that the control system is stable. Moreover, $\dot{\boldsymbol{e}}_\theta$ is bounded because $\boldsymbol{e}_\theta$, ε, $\tilde{\boldsymbol{\omega}}$, and $\hat{\varepsilon}_m$ are bounded. Then, we have

$$\lim_{t\to\infty}\int_0^t \frac{1}{2}\boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{Q}_\theta\boldsymbol{e}_\theta \mathrm{d}t \le V(0) - V(\infty) \le V(0). \tag{37}$$

By virtue to Barbalate's lemma, we reach the following conclusion:

$$\lim_{t\to\infty} \parallel \boldsymbol{e}_\theta \parallel = 0. \tag{38}$$

2. PD controller

In industrial control processes, the proportional-integral-derivative (PID) controller is one of the most vital and popular controllers. Its principle is based on eliminating the error from the previous error. According to different control requirements, various control structures have been presented, such as P, PI, and PD. In this work, a PD controller is employed to stably control the snake robot before the weights in CMAC converge. The PD controller is shown below:

$$u_{\mathrm{PD}} = k_{\mathrm{P}}(\bar\theta_{\mathrm{d}} - \bar\theta) + k_{\mathrm{D}}(\dot{\bar\theta}_{\mathrm{d}} - \dot{\bar\theta}) = k_{\mathrm{P}}\tilde\theta + k_{\mathrm{D}}\dot{\tilde\theta}, \tag{39}$$

where k_{P} and k_{D} are the proportional gain and derivative gain, respectively.

Remark 1 Stability can still be guaranteed with the proper design on the PD controller gains k_{P} and k_{D}.

Referring to Eq. (21), combining the PD controller with the perfect control law and substituting them into Eq. (20), we have

$$\begin{aligned} N\ddot{\bar\theta} &= N\ddot{\bar\theta}_{\mathrm{d}} + \lambda_{\mathrm{P}}\tilde\theta + \lambda_{\mathrm{D}}\dot{\tilde\theta} + Bk_{\mathrm{P}}\tilde\theta + Bk_{\mathrm{D}}\dot{\tilde\theta} \\ \Rightarrow \ddot{\tilde\theta} &= -\frac{\lambda_{\mathrm{P}} + Bk_{\mathrm{P}}}{N}\tilde\theta - \frac{\lambda_{\mathrm{D}} + Bk_{\mathrm{D}}}{N}\dot{\tilde\theta}. \end{aligned} \tag{40}$$

In this case, if we let $\bar k_{\mathrm{P}} = \frac{\lambda_{\mathrm{P}} + Bk_{\mathrm{P}}}{N}$ and $\bar k_{\mathrm{D}} = \frac{\lambda_{\mathrm{D}} + Bk_{\mathrm{D}}}{N}$ and make $s^2 + \bar k_{\mathrm{D}}s + \bar k_{\mathrm{P}}$ always Hurwitz stable via the proper design of k_{P} and k_{D}, then we can have the same conclusion as shown below:

$$\lim_{t\to\infty}\tilde\theta = 0. \tag{41}$$

Hence, the same stability analysis can be followed and it can be concluded that the overall control system is also stable.

3. Overall controller

In summary, the proposed overall motion controller can be rewritten as

$$u = \hat{\boldsymbol{\omega}}^{\mathrm{T}}\boldsymbol{\varphi} + \hat{\varepsilon}_m\mathrm{sign}(\boldsymbol{e}_\theta^{\mathrm{T}}\boldsymbol{P}_\theta\boldsymbol{B}_\theta) + \boldsymbol{K}_{\mathrm{PD}}\boldsymbol{e}_\theta, \tag{42}$$

where $\boldsymbol{K}_{\mathrm{PD}} = \begin{bmatrix}k_{\mathrm{P}}, & k_{\mathrm{D}}\end{bmatrix}^{\mathrm{T}}$ is the PD controller gain vector.

4 Numerical studies

This section describes the overall motion control system with the snake robot model as implemented and simulated in MATLAB/Simulink to validate the effectiveness and feasibility of the proposed control scheme with environmental uncertainties.

The parameters of the snake robot prototype are as follows: the mass is $m = 1$ kg, the number of links is $N = 8$, the length of each single link is $2l = 0.07$ m, and the moment of inertia is $J = 0.0016$ kg·m^2.

To simulate the various ground conditions, different friction coefficients of the contact surface are chosen in two cases: (1) $c_{\mathrm{t}} = 1$, $c_{\mathrm{n}} = 3$; (2) $c_{\mathrm{t}} = 2$, $c_{\mathrm{n}} = 6$. The total simulation time is 48 s for each case. To illustrate the steering control performance, a trapezoid wave is chosen as the desired trajectory. By following the desired trapezoid wave for the steering motion, the snake robot will move in an S-shape in the planar ground.

Figs. 11a and 11b demonstrate the robot tracking performance based on the proposed motion controller under cases 1 and 2, respectively.

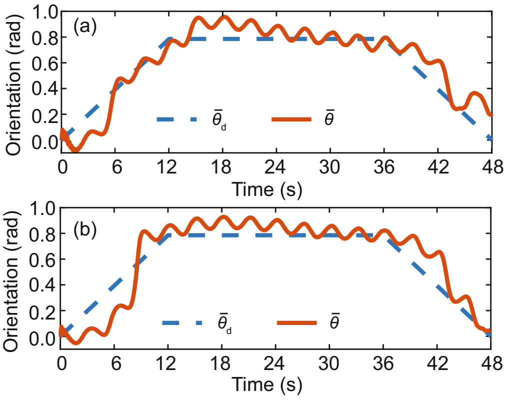

Fig. 11 Orientation outputs using the proposed controller while $c_t = 1$, $c_n = 3$ (a) or $c_t = 2$, $c_n = 6$ (b)

As can be seen from the figures, the snake robot follows the desired steering motion in both cases. This implies that the proposed control scheme can adapt to different environmental conditions.

A comparison study between the pure PD controller and the proposed motion controller is performed. Figs. 12a and 12b show the tracking errors using different controllers in different cases. Table 1 shows their root-mean-square errors (RMSEs) and mean absolute errors (MAEs).

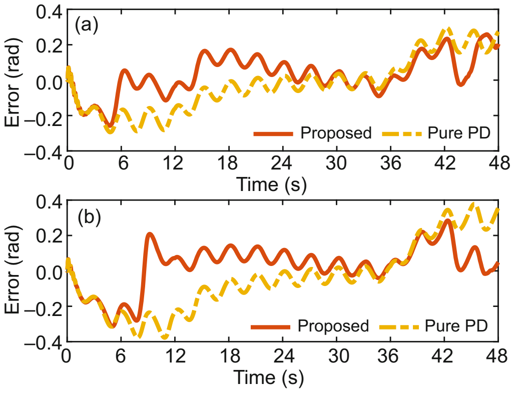

Fig. 12 Errors of the two different controllers when $c_t = 1$, $c_n = 3$ (a) or $c_t = 2$, $c_n = 6$ (b)

From these simulation results, the tracking errors obtained using the proposed motion controller are smaller than the ones obtained using the pure PD controller. The RMSEs are reduced by 28.4% and 37.5% respectively, while the MAEs are reduced by 31.5% and 40.0% respectively. Therefore, it can be concluded that the proposed motion controller has better steering control performance than the conventional controller.

Table 1 Errors of different controllers in different friction forces

Friction coefficients	Conroller	RMSE (rad)	MAE (rad)
$c_t = 1, c_n = 3$	Proposed	0.1087	0.0834
	Pure PD	0.1519	0.1218
$c_t = 2, c_n = 6$	Proposed	0.1180	0.0908
	Pure PD	0.1888	0.1513

In addition, note that the error differences of both RMSEs and MAEs while using the proposed controller in different cases are very small, both around 8% increasing from case 1 to case 2, whereas the friction coefficients are two times different between the two cases. On the other hand, these errors increase by about 19.5% from case 1 to case 2 while using the pure PD controller. This implies that the proposed motion controller can achieve better robustness than the conventional controller. This is attributed to the learning ability of the proposed CMAC.

In summary, it can be concluded from the simulation results that the proposed motion controller can achieve good steering control performance and good adaptability when subjected to environmental changes. The online learning ability of CMAC plays a crucial role in compensating for the environmental uncertainties. The proposed controller is also better than the pure PD controller in terms of tracking performance and robustness.

5 Experimental results

An experiment using the snake robot prototype shown in Fig. 2 is conducted to validate the effectiveness and feasibility of the proposed adaptive learning control scheme. The performance of the steering motion controller is investigated via the trajectory tracking experiment (Fig. 13).

During the experiment, a checkerboard paper is attached to a rough table, served as the experimental platform with varying surface frictions. In addition, for comparison, an experiment using the pure PD controller is performed.

Due to limited space, a ramp wave instead of a trapezoid wave (but with a turning rate similar to that in the simulation) is employed as the orientation reference for the snake robot. The snake robot is placed parallel to the x axis initially.

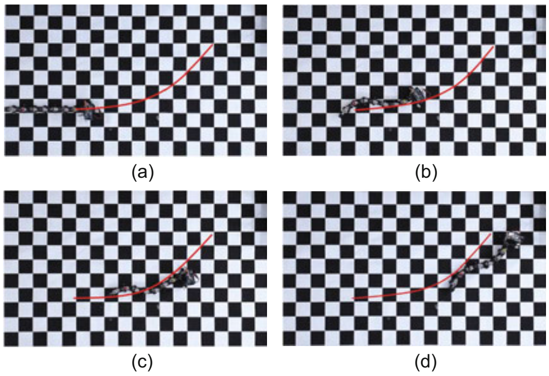

Fig. 13 A trajectory tracking experiment of the snake robot. (a) to (d) indicate the sequential photos during the steering motion control

Fig. 14 illustrates the experimental results while the proposed controller is applied. This shows that the snake robot can track the desired orientation successfully with small fluctuations.

To illustrate the advantage of the proposed motion controller, the comparison experimental results are shown in Fig. 15, which shows the tracking errors obtained using the proposed controller and the pure PD controller. It can be observed that both controllers can control the steering motion stably. However, the tracking performance of the proposed controller is significantly better than that of the pure PD controller.

Table 2 illustrates the RMSEs and MAEs using the different controllers in the experiment. It is obvious that the proposed motion controller can achieve smaller RMSEs and MAEs than the pure PD controller. The RMSE is reduced by 19.4% while the MAE is reduced by 17.4% with the use of the proposed controller.

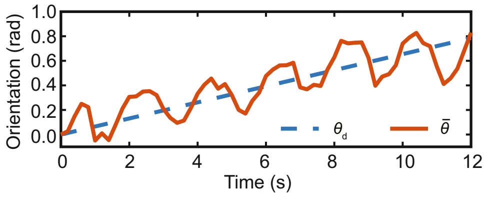

Fig. 14 Trajectory tracking of the snake robot

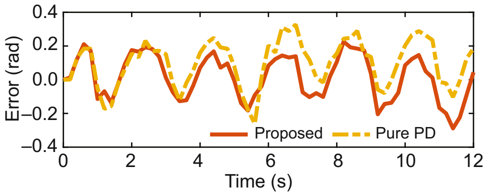

Fig. 15 Tracking errors between the two different controllers

Table 2 Errors in the experiment

Controller	RMSE (rad)	MAE (rad)
Proposed	0.1364	0.1182
Pure PD	0.1693	0.1431

Based on the results obtained from the above experiments, it can be concluded that the proposed motion controller can help the snake robot achieve good tracking performance and good adaptability in a varying environment. In addition, the comparison experiment indicates that the online learning ability of CMAC helps the control system improve the steering motion control performance.

6 Conclusions

In this study, a biomimetic control approach is proposed for steering a snake robot. In this hierarchical control scheme, a CPG-based locomotion generator is designed to generate the desired joint angle commands for the joint motors. Then a CMAC plus PD controller is proposed for orientation control of the snake robot, where CMAC acts as a kind of neural network controller inspired by the cerebellum. Both numerical and experimental studies show that the snake robot can be controlled successfully by the proposed approach. The pure PD controller (a conventional controller) is also applied in simulations and experiments for comparison. The comparison results show that the proposed motion controller can achieve better tracking performance and adaptability in a varying environment. In addition, this study presents a biomimetic way of controlling a bio-inspired robot, where the CPG-based locomotion generator and CMAC-based motion controller are combined to control the steering motion of the snake robot.

References

Albus JS, 1975. A new approach to manipulator control: the cerebellar model articulation controller (CMAC). *J Dynam Syst Meas Contr*, 97(3):220-227. https://doi.org/10.1115/1.3426922

Crespi A, Ijspeert AJ, 2008. Online optimization of swimming and crawling in an amphibious snake robot. *IEEE Trans Robot*, 24(1):75-87. https://doi.org/10.1109/TRO.2008.915426

Date H, Hoshi Y, Sampei M, 2000. Locomotion control of a snake-like robot based on dynamic manipulability. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.2236-2241. https://doi.org/10.1109/IROS.2000.895301

Gray J, 1946. The mechanism of locomotion in snakes. *J Exp Biol*, 23(2):101-120.

Guo X, Ma SG, Li B, et al., 2018. A novel serpentine gait generation method for snake-like robots based on geometry mechanics. *IEEE/ASME Trans Mech*, 23(3):1249. https://doi.org/10.1109/TMECH.2018.2809786

Hirose S, 1993. Biologically Inspired Robots: Snake-Like Locomotors and Manipulators. Oxford University Press, Oxford.

Hooper SL, 2000. Central pattern generators. *Curr Biol*, 10(5):R176-R179.

Inoue K, Ma SG, Jin CH, 2004. Neural oscillator network-based controller for meandering locomotion of snake-like robots. Proc IEEE Int Conf on Robotics and Automation, p.5064-5069. https://doi.org/10.1109/ROBOT.2004.1302520

Ito M, 2000. Mechanisms of motor learning in the cerebellum. *Brain Res*, 886(1-2):237-245.

Liljeback P, Haugstuen IU, Pettersen KY, 2012. Path following control of planar snake robots using a cascaded approach. *IEEE Trans Contr Syst Technol*, 20(1):111-126. https://doi.org/10.1109/TCST.2011.2107516

Marder E, Bucher D, 2001. Central pattern generators and the control of rhythmic movements. *Curr Biol*, 11(23):R986-R996. https://doi.org/10.1016/S0960-9822(01)00581-4

Mattia F, Paolo A, Luigi F, 2004. Bio-inspired Emergent Control of Locomotion Systems. World Scientific Publishing, Singapore.

Mukherjee J, Mukherjee S, Kar IN, 2017. Sliding mode control of planar snake robot with uncertainty using virtual holonomic constraints. *IEEE Robot Autom Lett*, 2(2):1077-1084. https://doi.org/10.1109/LRA.2017.2657892

Ohno H, Hirose S, 2001. Design of slim slime robot and its gait of locomotion. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.707-715. https://doi.org/10.1109/IROS.2001.976252

Poggio T, Girosi F, 1990. Networks for approximation and learning. *Proc IEEE*, 78(9):1481-1497. https://doi.org/10.1109/5.58326

Ryu JK, Chong NY, You BJ, et al., 2010. Locomotion of snake-like robots using adaptive neural oscillators. *Intell Ser Robot*, 3(1):1. https://doi.org/10.1007/s11370-009-0049-4

Tanaka M, Matsuno F, 2014. Modeling and control of head raising snake robots by using kinematic redundancy. *J Intell Robot Syst*, 75(1):53-69. https://doi.org/10.1007/s10846-013-9866-y

Wu XD, Ma SG, 2010. CPG-based control of serpentine locomotion of a snake-like robot. *Mechatronics*, 20(2):326-334. https://doi.org/10.1016/j.mechatronics.2010.01.006

Yu JZ, Tan M, Chen J, et al., 2014. A survey on CPG-inspired control models and system implementation. *IEEE Trans Neur Networks Learn Syst*, 25(3):441-456. https://doi.org/10.1109/TNNLS.2013.2280596

Zhang AF, Ma SG, Li B, et al., 2016. Adaptive controller design for underwater snake robot with unmatched uncertainties. *Sci China Inform Sci*, 59(5):052205. https://doi.org/10.1007/s11432-015-5421-8