Scientia Iranica B (2018) 25(5), 2675–2692

Sharif University of Technology
Scientia Iranica
Transactions B: Mechanical Engineering
http://scientiairanica.sharif.edu

Hierarchical decentralized control of a five-link biped robot

M. Yazdani, H. Salarieh*, and M. Saadat Foumani

School of Mechanical Engineering, Sharif University of Technology, Tehran, Iran.

Received 14 February 2017; received in revised form 11 June 2017; accepted 7 November 2017

KEYWORDS
Dynamic robot;
Biped walking;
Decentralized control;
Hierarchical control;
Online learning.

Abstract. Most of the biped robots are controlled using pre-computed trajectory methods or methods based on multi-body dynamics models. The pre-computed trajectory-based methods are simple; however, a system becomes highly vulnerable to the external disturbances. In contrast, dynamic methods make a system act faster, yet extensive knowledge is required about the kinematics and dynamics of the system. This fact gave rise to the main purpose of this study, i.e., developing a controller for a biped robot to take advantage of the simplicity and computational efficiency of trajectory-based methods and the robustness of the dynamic-based approach. To do so, this paper presents a two-layer hierarchical control framework for an under-actuated, planar, five-link biped robot model. The upper layer contains a centralized dynamic-based controller and uses all system sensory data to generate stable walking. The lower layer in this structure is a decentralized trajectory-based controller network, which learns how to control the system based on the upper layer controller output. When the lower controller fails to control the system, the upper layer controller takes action and makes the system stable. Then, when the lower layer controller gets ready, the control of the system will be handed to this layer.

1. Introduction

Robots and, especially, biped robots have been a research magnate in the last decades [1-8], giving rise to fascinating robots and products. To achieve such great products, a wide range of topics from studying biological locomotion and their mechanical model, model formulation, methods of gait synthesis, and the mechanical realization of biped robots to the control of such systems have been addressed in the literature, e.g., see [9-14].

Recent studies have shown that the control of locomotion in mammals, including humans, is based on the neural circuits activities within the spinal cord (the Central Pattern Generator, CPG) [15-17]. Furthermore, these circuits are largely responsible for learning rhythmic activities. In other words, through conscious training, the learned motor patterns seem to act as automatic and ingrained as CPG-driven motor patterns clearly do [18]. This means that after the circuits learn the pattern, the locomotion voluntarily starts by the brain. Then, the circuits control the muscle activation, making locomotion possible. This implies that there exists a low-level programming hierarchy that allows rhythmic activities without the involvement of the brain [19].

This idea is the backbone of the current study. In other words, the main objective of this article is to develop a hierarchical two-layer controller framework for a walking biped robot which replicates the function of the brain in the learning process at the upper layer

*. *Corresponding author. Tel.: +98 21 66165538; Fax: +98 21 6600 0021*
E-mail addresses: masoudyazdani@mech.sharif.ir (M. Yazdani); salarieh@sharif.ir (H. Salarieh); m saadat@sharif.ir (M. Saadat Foumani)

doi: 10.24200/sci.2017.4588

and the function of the neural circuits in the control of locomotion at the lower layer. In this framework, by connecting controllers of the lower layer in a chain structure, they have the ability to generate and control the desired locomotion without any prior knowledge about the system. Furthermore, its distributed nature makes it more robust to communicate and deal with hardware failure. Besides, this framework can be used to reduce production costs and increase modularity in the robots' designs. At last, by using a sophisticated high-level controller as a supervisory controller, the system will take advantage of the robustness and stability of the high-level controller as well as the simplicity and computation efficiency of low-level controllers. It should be noted that the concept of hierarchical controller and estimation is used for a variety of applications. For more information, see [20-30].

The high-level controller in this framework controls the robot using dynamic-based methods, as shown in [31-34]. In these strategies, the main focus is on the dynamics and kinematics of the system to generate gait motion. Although such an approach needs extensive knowledge of the mechanical structure and high computational power, they are performed more robustly than trajectory-based controllers under disturbances.

The low-level controllers in this framework represent a network of simple trajectory-based controllers which operate based on the pre-calculated joint trajectories. In this approach, the reference trajectories are pre-computed, and then, these trajectories are pursued by means of feedback controllers. To compute these trajectories, some researchers used optimization of various cost functions over a walking cycle, as in [35]; and some others extracted them from their analogy with biological or simpler mechanical systems [36]. Obviously, trajectories in the form of time value functions are not suitable to be used as controllers' tracking input signals. To solve this problem, inspired by neural circuits in the spinal cord, stable oscillators are used to encode these signals into parameters of nonlinear oscillators with an intrinsic limit cycle property [37-41]. Moreover, the structure of these oscillators supports the stability of the overall system via feedback integration and offers a solid foundation for learning and optimization algorithms [42].

In Section 2, the dynamic model of a five-link biped robot corresponding to "RABBIT" prototype is derived. In this model, it is assumed that the double support phase is instantaneous, and the impact is modelled as a contact between rigid bodies. Furthermore, it is assumed that the contact between the stance leg and the ground acts as a pivot during the single support phase. Then, the controller structure for this robot is proposed in Section 3. As mentioned before, this structure has two independent layers. At the upper layer, the controller is a sophisticated dynamic-based controller which uses all available sensory data from the system to generate stable movement. For the considered robot, this controller is developed based on partial feedback linearization method for under-actuated systems [43]. In order to do that, a set of outputs in terms of robot configuration is considered so that nullifying the outputs makes the system have the desired posture [32]. This controller is discussed in Section 3.1. On the other hand, at the lower level of the proposed structure, the system is controlled by a network of simple trajectory-based controllers. The structure of this network and its nodes will be presented in Section 3.2. After discussing the control framework and its components, its implementation on the considered robot is simulated, and the results are presented in Section 4. Finally, the conclusions is drawn in Section 5.

2. Robot model

The model used in this paper is based on a prototype, called "RABBIT", developed by *Le Centre national de la recherche scientifique*, Paris, France [44].

This model consists of a link, which represents the torso, and two identical kinematic open chains made up of two links with pointed ends, which represent the legs. These links are pivoted together at a point called hip (Figure 1(a)). Therefore, the model has five DoFs. Moreover, at each joint, a torque exerting actuator is used as one of the inputs; thus, the system has four independent inputs. In this model, it is assumed that the motion is confined to sagittal plane, and the walking of the robot is considered on the surface level. Besides this assumption, the walking of the robot is interpreted as consecutive single support phases (meaning only one leg is on the ground and acts as a pivot) and transition from one leg to another takes place in an infinitesimal length of time [31].

Therefore, the model of the robot can be described by two parts:

1. A differential equation representing the governing dynamics during single support phases;
2. An impulse model representing the contact event (modelled as a contact between rigid bodies).

2.1. Single support phase model

During the single support phase, the stance leg acts as a pivot. Therefore, the dynamic model of the robot during this phase has five DoFs. Let $\mathbf{q} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)^T$ be the set of coordinates depicted in Figure 1(a); $\mathbf{u} = (u_1, u_2, u_3, u_4)^T$ represents the inputs of the system presented in Figure 1(c). Since only symmetric gaits are of interest, the same model

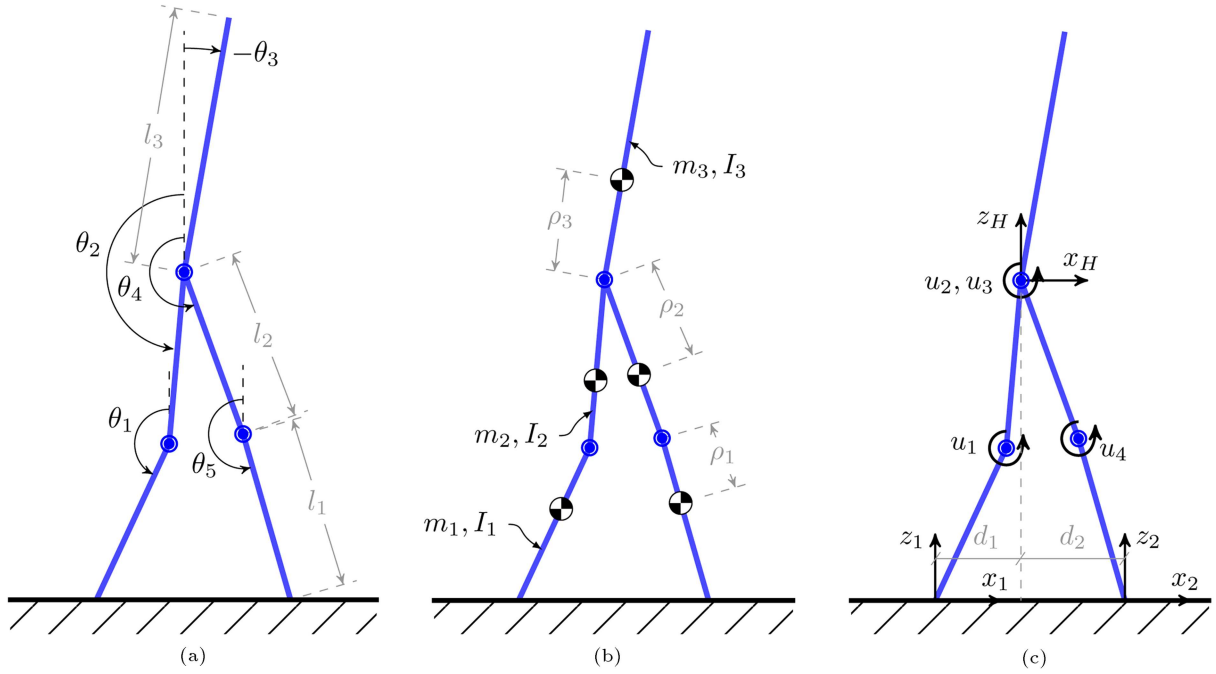

Figure 1. Schematic of biped robot: (a) All the absolute angles and the lengths of the links, (b) all the masses and centers of masses, and (c) all the inputs.

can be used irrespective of which leg is the stance leg if the coordinates are relabeled after each impact [45]. By using the Lagrange method, the equation of motion for the robot can be derived as follows:

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{B}(\mathbf{q})\mathbf{u}, \tag{1}$$

where $\mathbf{D}(\mathbf{q})$, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$, and $\mathbf{B}(\mathbf{q})$ are inertia, Coriolis, and the inputs gain matrices, respectively. Also, $\mathbf{g}(\mathbf{q})$ represents the gravity forces acting on the joints. By writing the equation of motion in the state-space form, we have:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{q}} \\ \mathbf{D}^{-1}(\mathbf{q})(\mathbf{B}(\mathbf{q})\mathbf{u} - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{g}(\mathbf{q})) \end{bmatrix}$$
$$=: \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \tag{2}$$

where $\mathbf{x} = (\mathbf{q}^T, \dot{\mathbf{q}}^T)^T$.

2.2. Impact model

The impact between the swing leg and the ground is modelled as a contact between two rigid bodies. At the contact event, the following conditions are assumed [46]:

1. All joints are assumed perfectly elastic without any backlash;
2. The impact is instantaneous. The impulsive forces due to the impact may result in an instantaneous change in the velocities; however, there is no instantaneous change in the angles and positions;
3. The contact of the swing leg with the ground results in no rebound or no slipping of the swing leg, and the stance leg lifts from the ground without interaction.

From the first and second hypotheses, one can conclude that the angular momentum is preserved at the impact about the impact point. The equation of angular momentum conservation is solved with the equation derived from the third hypothesis. It yields an expression for the velocities of the links after the impact in terms of the velocities just before the impact. In other words, the impact model results in a smooth discrete map [32]:

$$\mathbf{x}^+ = \Delta(\mathbf{x}^-), \tag{3}$$

where $\mathbf{x}^-$ is the value of the states just before the impact and $\mathbf{x}^+$ is the value of the states just after the impact. Furthermore, function Δ is responsible for the mapping from the states of the system before impact to the states of the system after the impact.

2.3. Overall model

The overall dynamic system can be expressed as a hybrid system [32]:

$$\Sigma : \begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))\mathbf{u}(t) & \mathbf{x}^-(t) \notin \mathcal{S} \\ \mathbf{x}^+(t) = \Delta(x^-(t)) & \mathbf{x}^-(t) \in \mathcal{S}, \end{cases} \tag{4}$$

where $\mathcal{S}$ is the set of all feasible states belonging to the walking surface defined as follows:

$$\mathcal{S} := \{(\mathbf{q}, \dot{\mathbf{q}}) | z_2 = 0, x_2 > 0\}, \tag{5}$$

where x_2 and z_2 are the end tip locations of the swing leg (see Figure 1(c)). In other words, the trajectory evolution of the system is described by Eq. (2) until the impact occurs (when the states of the system belong to $\mathcal{S}$). The impact, which is described by Eq. (3), changes the states instantaneously and relabels them to be used as the next single support phase initial condition.

3. Control architecture

The main purpose of the presented architecture is to use the robustness of a dynamic-based controller and the simplicity and computation efficiency of trajectory-based techniques at the same time in the control of biped robots. Hence, as illustrated in Figure 2, it is proposed that this architecture contains two entities:

1. A high-level controller which is a dynamic based controller;
2. A network of low-level controllers in which each node is a simple trajectory tracking controller that corresponds to an actuator of the robot.

This method works as follows: first, the system is controlled by the high-level controller. Meanwhile,

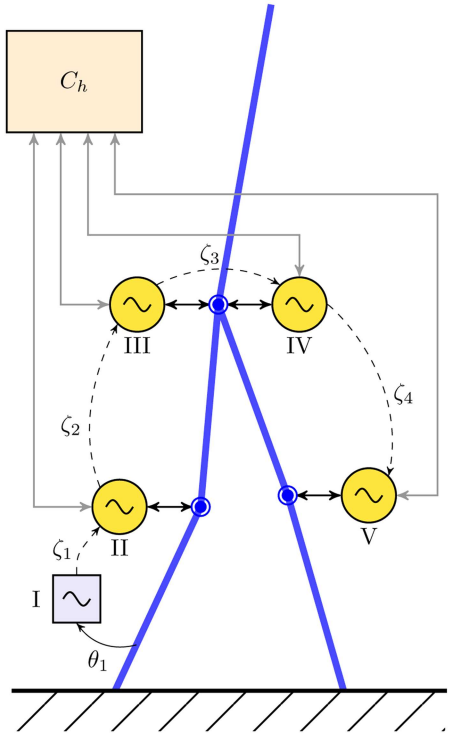

Figure 2. Schematic of low-level controllers network and their interaction with the high-level controller (C_h); the yellow circles with the wave sign in the middle represent controllers' nodes and the blue square with the wave sign inside is the representation of the feedback node; ζ_i for $i = 1 \cdots 4$ is the output of the ith low-level controller which synchronizes the $(i+1)$th node with the ith node (see Section 3.2.2).

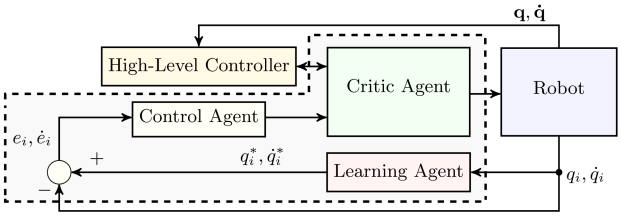

Figure 3. Controller's node structure in interaction with the robot and the high-level controller; the dashed line border area is the low-level controller boundary.

the input of each actuator (generated by the high-level controller) is fed into the corresponding low-level controller. This signal is used by each node to learn to reproduce the desired trajectory of the corresponding DoF. At the same time, each node evaluates whether it can generate the desired trajectory or not. Next, if all nodes in the network have the ability to generate the desired trajectories, the high-level controller gets turned off and the robot will be controlled by low-level controllers. A schema of this process is shown in Figure 3.

Obviously, when the system is controlled by the network of the low-level controllers, it should be stable and imitate the output of the high-level controller. Hence, due to its decentralized nature, each node should generate the output in synchronization with the unactuated DoF. Therefore, the nodes are connected together in a chain structure and the leader in this chain is synchronized with the unactuated DoF using a feedback node (Figure 2). The feedback node structure is discussed thoroughly in Section 3.2.2. Furthermore, if the system is disturbed in this state, preventing any of the nodes from tracking its desired trajectory, the high-level controller gets turned on and controls the system.

3.1. High-level controller

This controller design is based on the proposed method by [31]. The fundamental idea of this controller is to encode walking in terms of a set of posture conditions, which are in turn expressed as holonomic constraints on the position variables. These virtual constraints are then used to construct outputs of the model and are imposed on the robot via a feedback control. For the considered simple model, the following constraint has been chosen [32]:

$$\mathbf{y} = \mathbf{K} \cdot \mathbf{h}(\mathbf{q}), \tag{6}$$

where:

$$\mathbf{h} := \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{bmatrix} = \begin{bmatrix} (\theta_3 - \theta_{3d}) \\ (d_1 + d_2) \\ (z_H - z_{Hd}(d_1)) \\ (z_2 - z_{2d}(d_1)) \end{bmatrix}, \tag{7}$$

and $\mathbf{K}$ is a diagonal matrix defined as follows:

$$\mathbf{K} = \begin{bmatrix} k_1 & 0 & 0 & 0 \\ 0 & k_2 & 0 & 0 \\ 0 & 0 & k_3 & 0 \\ 0 & 0 & 0 & k_4 \end{bmatrix}. \quad (8)$$

It should be noted that in Eq. (7):

$$\begin{aligned} x_1 &= 0, \\ z_1 &= 0, \\ x_H &= l_1 \sin(\theta_1) + l_2 \sin(\theta_2), \\ z_H &= -l_1 \cos(\theta_1) - l_2 \cos(\theta_2), \\ x_2 &= x_H - l_1 \sin(\theta_5) - l_2 \sin(\theta_4), \\ z_2 &= z_H + l_1 \cos(\theta_5) + l_2 \cos(\theta_4), \\ d_1 &= x_H - x_1 = l_1 \sin(\theta_1) + l_2 \sin(\theta_2), \\ d_2 &= x_H - x_2 = l_1 \sin(\theta_5) + l_2 \sin(\theta_4). \end{aligned} \quad (9)$$

As illustrated in Figure 1(c), (x_H, z_H) and (x_2, z_2) are the hip and swing foot-end locations with respect to the location of stance foot-end, i.e., (x_1, z_1). Moreover, in Eq. (8), gains k_1, k_2, k_3, and k_4 are scaling constants. Note that the latter constants normalize the outputs in the process of controller design.

Satisfying the first constraint implies that the robot torso should maintain its desired constant angle (i.e., θ_{3d}), the second one makes the robot take steps and advance its hip, and the last two control trajectories of the hip and swing leg foot-end in order to have quite natural walking gait. To do so, the desired trajectories are defined as second-order polynomial with respect to the horizontal location of the hip (i.e., d_1) such that [32]:

$$\begin{aligned} z_{Hd}(\pm \text{ sld}/2) &= z_{H\min}, \\ z_{Hd}(0) &= z_{H\max}, \end{aligned} \quad (10)$$

and:

$$\begin{aligned} z_{2d}(\pm \text{ sld}/2) &= 0, \\ z_{2d}(0) &= z_{2\max}, \end{aligned} \quad (11)$$

where sld is the desired step length, $z_{H\max}$ and $z_{H\min}$ are the maximum and minimum desired values of z_H over a step, and $z_{2\max}$ is the maximum desired value of z_2 over a step.

To design a controller to satisfy constraints of Eq. (6)(drive them to zero), their governing dynamics should be derived. To do so, time derivatives of constraints are considered. Obviously, the dynamics of the system is a second-order one, and constraints are independent of the generalized velocities. Therefore, the first derivative of the constraints dynamics is not explicitly dependent on the inputs of the system. However, the second derivative of the constants dynamics depends on the inputs of the system and can be written as follows:

$$\frac{d^2\mathbf{y}}{dt^2} = L_f^2\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) + L_g L_f \mathbf{h}(\mathbf{q})\mathbf{u}, \quad (12)$$

where $L_g L_f \mathbf{h}(\mathbf{q})$ is called the decoupling matrix, and $L_f \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}})$ is the Lie derivative of $\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}})$ along the vector field of $\mathbf{f}(\mathbf{q}, \dot{\mathbf{q}})$. If the decoupling matrix is invertible, which is true in the region of interest [32], one can choose the following controller:

$$\mathbf{u} = (L_g L_f \mathbf{h})^{-1}(-L_f^2\mathbf{h} + \mathbf{v}). \quad (13)$$

In this case, the governing dynamic equation of the closed-loop system becomes:

$$\ddot{\mathbf{y}} = \mathbf{v}. \quad (14)$$

Therefore, in order to stabilize the virtual constraints, the following PD feedback is used:

$$\mathbf{v} = \kappa_1.\mathbf{y} + \kappa_2.\dot{\mathbf{y}}, \quad (15)$$

where $\kappa_1 < 0$ and $\kappa_2 < 0$. This feedback structure makes the origin in the space of the constraints exponentially stable. Subsequently, the trajectory of the system is a stable periodic orbit [31], which means that the robot is walking stably.

3.2. Low-level controller

The network of low-level controllers consists of four controller nodes and a feedback node. Each controller node is attached to a joint of the robot (do not relabel with the change of stance leg) and the feedback node is attached to the stance-leg shank (see Figure 2). Therefore, it is very easy to work in relative coordinate $\bar{\mathbf{q}} := (q_a, \mathbf{q}_b)^T$ where $q_a = \pi - \theta_1$ is an unactuated coordinate, and $\mathbf{q}_b = (\psi_1, \psi_2, \psi_3, \psi_4)^T$ represents actuated coordinates equal to $(\theta_2 - \theta_1, \theta_4 - \theta_5, \theta_2 + \theta_3, \theta_4 + \theta_3)^T$ when leg number one is the stance leg and equal to $(\theta_4 - \theta_5, \theta_2 - \theta_1, \theta_4 + \theta_3, \theta_2 + \theta_3)^T$ when leg number two is the stance leg.

Therefore, the dynamic model in Eq. (1) in the coordinates used in low-level controllers network can be written as follows:

$$\bar{\mathbf{D}}(\bar{\mathbf{q}})\ddot{\bar{\mathbf{q}}} + \bar{\mathbf{C}}(\bar{\mathbf{q}}, \dot{\bar{\mathbf{q}}})\dot{\bar{\mathbf{q}}} + \bar{\mathbf{g}}(\bar{\mathbf{q}}) = \bar{\mathbf{B}} \cdot \mathbf{u}, \quad (16)$$

where it can be easily shown that $\bar{\mathbf{B}}$ has the form of $\bar{\mathbf{B}} = [\mathbf{0}_{4\times 1}, \mathbf{I}_{4\times 4}]^T$.

In the rest of this section, the components of the low-level controller are discussed. As depicted in Figure 3, each controller node in the low-level controllers network has three components as follows:

- Learning agent whose mission is to reproduce the desired trajectory synchronously with the desired trajectories of the other DoFs based on the outcomes of the system when the system is controlled by the high-level controller. This agent and its structure are discussed in Section 3.2.2;
- Critic agent is in charge of switching between the high-level controller and low-level controllers based on the control eligibility of low-level controllers. This agent is introduced and discussed in Section 3.2.3;
- Control agent which is to make sure that the system follows the desired trajectories. This agent and its properties are brought up in Section 3.2.1.

3.2.1. Control agent

Suppose that the system is controlled by the high-level controller and its trajectory is a stable limit-cycle. Furthermore, assume that the desired trajectory and its corresponding input signals are generated by low-level controllers within a satisfactory error, i.e., the generated trajectory is expressed as $(\bar{\mathbf{q}}^*, \dot{\bar{\mathbf{q}}}^*)$, and evolution of the desired input signals is denoted by $\mathbf{u}^*$. These conditions make the system switch be controlled by low-level controllers.

When the system is controlled by the low-level controller, the control agents try to eliminate the difference between the desired and actual trajectories. To do so, the control agent output for the ith actuated DoF is considered as DoF is considered as follows:

$$u_i = u_i^* + \delta u_i, \tag{17}$$

where δu_i should stabilize the perturbed system from the desired trajectory. It means that the designed controller makes the system follow the desired trajectory. To design the controller, let us consider the perturbation from the desired trajectory which is denoted by $(\delta\bar{\mathbf{q}}, \delta\dot{\bar{q}})$. The governing equation for the ith perturbed actuated DoF can be rewritten as follows:

$$\bar{D}_{ii}\delta\ddot{\bar{q}}_i + \Delta_i^{(1)}\delta\dot{\bar{q}}_i + \Delta_i^{(2)}\delta\bar{q}_i + \Delta_i^{(3)} = \delta u_i, \tag{18}$$

where $\Delta_i^{(1)}$, $\Delta_i^{(2)}$, and $\Delta_i^{(3)}$ are given in Appendix A.

Due to the existence of a critic agent, the large disturbances make the system use the high-level controller instead of the network of the low-level controllers. This means that all variables are bounded on the domain of a low-level controller definition. Hence, these bounds can be used in the stability analysis and controller design for the low-level controllers. In order to design a controller to make the system follow the desired trajectory, consider the following proposition.

Proposition 1. *The high-gain feedback structure in Eq. (19) for δu_i makes the system in Eq. (18) stable.*

$$\delta u_i = -\frac{1}{\epsilon}\left(\delta\dot{\bar{q}}_i + c\delta\bar{q}_i\right), \tag{19}$$

where $c > 0$ and $\epsilon \ll 1$.

Proof. The closed-loop equation for the ith actuated DoF with the proposed controller structure (19) can be written as follows:

$$\epsilon\bar{D}_{ii}\delta\ddot{\bar{q}}_i + \left(1 + \epsilon\Delta_i^{(1)}\right)\delta\dot{\bar{q}}_i + \left(c + \epsilon\Delta_i^{(2)}\right)\delta\bar{q}_i + \epsilon\Delta_i^{(3)} = 0, \tag{20}$$

where $\Delta_i^{(1)}$, $\Delta_i^{(2)}$, and $\Delta_i^{(3)}$ are defined in Appendix A. Therefore, by using singular perturbation analysis [47] (see Appendix B), the closed-loop response will be as follows:

$$\begin{aligned}\delta\bar{q}_i(t) &= \delta\bar{q}_i(0)e^{-ct} + \mathcal{O}(\epsilon),\\ \delta\dot{\bar{q}}_i(t) &= -c\delta\bar{q}_i(0)e^{-ct} + (\delta\dot{\bar{q}}_i(0) + c\delta\bar{q}_i(0))\delta_D(t/\epsilon) + \mathcal{O}(\epsilon),\end{aligned} \tag{21}$$

where $\delta_D(\cdot)$ decays exponentially with the rate of its argument and $\delta_D(0) = 1$. This response means that the closed-loop system in Eq. (18) with controller in Eq. (19) is stable. □

Proposition 1 implies that the controller agent with structures (17) and (19) guarantees that the ith actuated DoF will achieve a trajectory within $\mathcal{O}(\epsilon)$-neighbor of the desired trajectory.

3.2.2. Imitating/learning agent

The missions of this agent are learning and generating the desired trajectory of its corresponding DoF, which is a periodic motion. Therefore, this agent is essentially a dynamical system with learning capability to reproduce frequency contents of the desired trajectory. In order to regenerate frequency content of a dynamical system, based on the following theorem, a new oscillator-based model is presented.

Theorem 2. *Consider the following system:*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{22}$$

where $\mathbf{x} \in \mathbb{R}^n$ represents the states of the system. Assume that system in Eq. (22) has a self-sustained oscillatory stable trajectory with a period of T. Then, for any arbitrary integer, $0 < n_1 \leq n$, there exists a transformation, $\mathbf{x} = \mathbf{T}(\xi, \eta)$, which transforms the local dynamics of the system in the neighborhood of the closed-loop trajectory into the following equations:

$$\dot{\xi} = \mathbf{A}(\xi, \eta), \tag{23}$$

$$\dot{\eta} = \mathbf{B}(\xi, \eta), \tag{24}$$

where the system in Eq. (23) is a neutral system (it is neither stable nor unstable) whose state $\xi \in \mathbb{R}^{n_1}$ shows

monotonic behavior and $\mathbf{A}(\xi + T, 0) = \mathbf{A}(\xi, 0)$. *In addition, the system in Eq. (24) with state* $\eta \in \mathbb{R}^{n-n_1}$ *has an asymptotically stable equilibrium point* $\eta = \mathbf{0}$ *mapped onto the aforesaid oscillatory trajectory of the system in Eq. (22).*

Proof. See Appendix C. □

Due to the second-order nature of the biped, it can be concluded that the desired trajectory of each DoF is the projection of the system trajectory into the state space of the corresponding DoF. Therefore, by using Theorem 2, to reproduce the desired trajectory corresponding to a DoF, it is enough to have a monotonic dynamical system with periodic time derivative and a map from this state to the desired trajectory in the state space of the corresponding DoF. Interestingly, this idea has been used previously to model the human locomotion based on the cadence and electromyographic data obtained from a human subject who walked on a treadmill [48].

To implement this idea, a structurally stable oscillator is used as the aforesaid monotonic dynamical system. Then, an approximate function is used to map the oscillator output to the desired output signal. This function is called *Shaping Network*. It should be noted that the inputs of the Shaping Networks are the phases of their corresponding oscillators. This structure is depicted in Figure 4. This structure has two working modes: *Learning* and *Imitating* modes; they are discussed in the following paragraphs.

At the learning phase, the desired oscillatory signal is fed into the oscillator and neural network. Meanwhile, the base frequency of the input signal will be learned by the algorithm discussed in Section 3.2.2. In addition to that, the phase difference between connected oscillators of the imitating agents of low-level controllers will be learned by the algorithm discussed in Sections 3.2.2. Simultaneously, based on the output phase of the oscillator and desired trajectory, the shaping network will be trained in such a way that the error between the desired trajectory and the output of the shaping network tends to zero. The structure of shaping network and the learning algorithm used have been discussed in Section 3.2.2.

At the imitating phase, training of the shaping network is finished, and it generates the desired signals based on the frequency and the phase of its input oscillator. Moreover, the connected oscillators will get synchronized together in the network of low-level controllers with the phase differences learned at the learning stage. The used synchronization algorithm will be discussed in Section 3.2.2. It should be noted that this synchronization is mandatory because projections of the system oscillation phase into the state spaces of actuated DoFs always are mapped onto the consistent phases of oscillators. This means that if one of the oscillators is disturbed, all oscillators should be disturbed synchronously, guaranteeing that the point generated by the outputs of all DoFs' imitating agents always lies in the desired trajectory.

When the system is fully actuated, i.e., the system does not exhibit internal dynamics, the aforesaid structure for learning and imitating rhythmic movement is sufficient. However, when the system is underactuated, the sum of dimensions of actuated subspaces is less than the order of the system. Therefore, the trajectory of the system cannot be fully expressed by its projections on the state spaces of actuated DoFs. In other words, states of internal dynamics of the system influence the overall system oscillation phase. In this case, in order to generate the desired trajectory, the phases of oscillators should be synchronized with the oscillation phase of the internal dynamics of the system. Hence, the projection phase of the system trajectory on the internal dynamics subsystem should be extracted. To extract the phase of the internal dynamics of the system, the structure depicted in Figure 5 is used. In this structure, based on internal dynamics of the

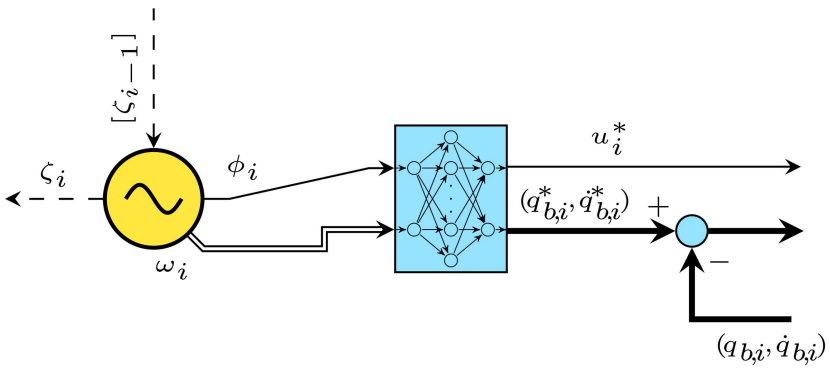

Figure 4. The considered structure of the imitating agent for each actuated DoF; the circle with a sinus sign in the middle is a representation of a dynamically stable oscillator, and the box with a symbol of a network in the middle is the representation of shaping network. Dashed lines show the synchronization signals transmitted to and received from the network of imitating agents of low-level controllers.

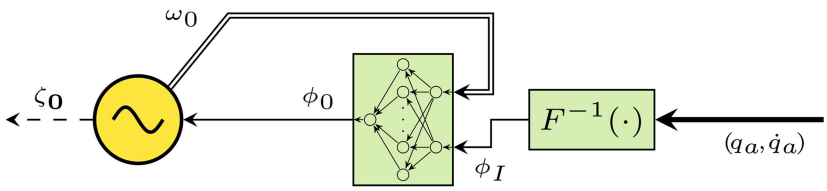

Figure 5. The considered structure of imitating agent for extracting internal dynamics phase; the circle with a sinus sign in the middle is a representation of a dynamically stable oscillator; $F^{-1}(\cdot)$ block represents the transformation function of internal dynamics states to the internal dynamics phase, and the box with a symbol of a network in the middle is a representation of shaping network which transforms internal dynamics phase to the phase of its base frequency component. Dashed lines show the synchronization signals transmitted to the network of imitating agents of low-level controllers.

system, the phase of trajectory projection on this subspace is extracted, and a stable oscillator will be synchronized to the base frequency component of this phase. Then, imitating agents' oscillators of the control nodes will be synchronized to the oscillator of this node, which is called the feedback node. It should be noted that the base frequency of the feedback node's oscillator is twice the base frequency of the system due to the relabeling of the legs at the impacts.

Oscillator structure. In order to train the shaping networks in the learning mode and reproduce the desired trajectory in the imitating mode, the input training signals, which are the phases of the oscillators (the outputs of the oscillators), should be oscillatory, and their intrinsic frequencies should be equal to the base frequency of the desired trajectory at the end of the learning phase and in the imitating phase.

In this manuscript, Hopf oscillators are used in order to generate the input signals of the shaping networks. A Hopf oscillator has a limit-cycle with circular symmetry on a two-dimensional plane. Its governing dynamics with an entrainment signal is expressed by the following differential equations [49]:

$$\dot{\mathbf{z}}_i = \begin{bmatrix} \dot{z}_{i,1} \\ \dot{z}_{i,2} \end{bmatrix} = \mathbf{F}(\mathbf{z}_i; \omega_i) + \varepsilon \begin{bmatrix} P_i \\ 0 \end{bmatrix} = \begin{bmatrix} \gamma(\mu^2 - z_{i,1}^2 - z_{i,2}^2) z_{i,1} - \omega_i z_{i,2} \\ \gamma(\mu^2 - z_{i,1}^2 - z_{i,2}^2) z_{i,2} + \omega_i z_{i,1} \end{bmatrix} + \varepsilon \begin{bmatrix} P_i \\ 0 \end{bmatrix}, \quad (25)$$

where μ controls the amplitude of the oscillations, $\gamma > 0$ controls the rate of convergence, and ω_i is the intrinsic frequency of the oscillator. In this oscillator, without perturbations (when $\varepsilon = 0$), the system is oscillating at ω_i rad s^{-1}. In order to adapt the intrinsic frequency to the base frequency of the desired trajectory, it is perturbed with an entrainment signal (P_i) which is proportional to the desired trajectory. Then, by using the following adaptation rule, the intrinsic frequency will converge to one of the input signal frequency component [49]:

$$\dot{\omega}_i = -\varepsilon P_i \frac{z_{i,2}}{\sqrt{z_{i,1}^2 + z_{i,2}^2}}. \quad (26)$$

At last, it should be noted that the output of the ith oscillator depicted in Figure 4 is the phase of oscillator, which is defined by the following equation:

$$\phi_i = \text{atan}\left(\frac{z_{i,2}}{z_{i,1}}\right). \quad (27)$$

Oscillators synchronization. As mentioned, it is mandatory that all oscillators be synchronized together in the imitating phase. To make them synchronized, consider the following theorem.

Theorem 3. *Consider the following network of n coupled Hopf oscillators in a chain structure:*

$$\begin{cases} \dot{\mathbf{z}}_0 = \mathbf{F}(\mathbf{z}_0; \omega_0) \\ \dot{\mathbf{z}}_1 = \mathbf{F}(\mathbf{z}_1; \omega_1) + k_s(\mathbf{R}_{\Delta\phi_{01}} \zeta_0 - \mathbf{z}_1) \\ \vdots \\ \dot{\mathbf{z}}_n = \mathbf{F}(\mathbf{z}_n; \omega_n) + k_s(\mathbf{R}_{\Delta\phi_{(n-1)n}} \zeta_{n-1} - \mathbf{z}_n) \end{cases} \quad (28)$$

where ζ_i is the output of the ith oscillator which is defined as follows:

$$\zeta_i(t) = \mathbf{z}_i\left(\frac{\omega_{i+1}}{\omega_i} t\right). \quad (29)$$

$\mathbf{R}_\theta$ is a θ planar rotation matrix which is defined as follows:

$$\mathbf{R}_\theta = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}, \quad (30)$$

and $\mathbf{F}(\cdot;\cdot)$ is the governing dynamics function of the Hopf oscillator defined in Eq. (25). Furthermore, it is assumed that $\omega_{i+1} \leq \omega_i$. In addition, $\Delta\phi_{ij}$ is the desired phase difference between the ith oscillator and the jth oscillator in the chain, and k_s is the synchronization gain.

Then, for any $k_s > \gamma\mu^2$, these n systems will reach synchrony exponentially regardless of the initial conditions. In this situation, the following relations will be held as time tends to infinity:

$$\begin{cases} \mathbf{z}_1 = \mathbf{R}_{\Delta\phi_{01}}\zeta_0 \\ \mathbf{z}_2 = \mathbf{R}_{\Delta\phi_{12}}\zeta_1 \\ \quad \vdots \\ \mathbf{z}_n = \mathbf{R}_{\Delta\phi_{(n-1)n}}\zeta_{n-1} \end{cases}. \tag{31}$$

Proof. Function $\mathbf{F}(\mathbf{z}_i, \omega_i) - k_s\mathbf{z}_i$ is contracting in Eq. (28) for any $k_s > \gamma\mu^2$ (see Appendix D), and Eq. (31) is a particular solution of the system in Eq. (28). Therefore, all subsystems of Eq. (28) will reach synchrony regardless of the initial conditions [50] and, in this case, Eq. (31) will be held. □

As a result, by using the following network, all oscillators of controller nodes will get synchronized to the leading oscillator of the chain (oscillator of the feedback node) with the desired phase delay:

$$\begin{cases} \dot{\mathbf{z}}_0 = \mathbf{F}(\mathbf{z}_0; 2\omega) \\ \dot{\mathbf{z}}_1 = \mathbf{F}(\mathbf{z}_1; \omega) + k_s(\mathbf{R}_{\Delta\phi_{01}}\zeta_0 - \mathbf{z}_1) \\ \dot{\mathbf{z}}_2 = \mathbf{F}(\mathbf{z}_2; \omega) + k_s(\mathbf{R}_{\Delta\phi_{12}}\zeta_1 - \mathbf{z}_2) \\ \dot{\mathbf{z}}_3 = \mathbf{F}(\mathbf{z}_3; \omega) + k_s(\mathbf{R}_{\Delta\phi_{23}}\zeta_2 - \mathbf{z}_3) \\ \dot{\mathbf{z}}_4 = \mathbf{F}(\mathbf{z}_4; \omega) + k_s(\mathbf{R}_{\Delta\phi_{34}}\zeta_3 - \mathbf{z}_4) \end{cases} \tag{32}$$

where:

$$\zeta_0(t) = \mathbf{z}_0\left(\frac{t}{2}\right),$$

$$\zeta_i(t) = \mathbf{z}_i(t), \qquad i \in \{1, 2, 3\}. \tag{33}$$

Learning phase differences between oscillators. When the system is in training mode, based on Eq. (25), all oscillators get synchronized with their input signals. Obviously, due to phase differences between input signals, it is expected that each oscillator has a phase difference with its neighbors in the chain. Therefore, to synchronize all oscillators together, phase differences between neighbors should be adaptively derived and learned in this mode. To do so, the following adaptive law is used to update this value in the training mode [51]:

$$\Delta\dot{\phi}_{i(i+1)} = -\gamma_\phi \sin\left(\frac{\omega_{i+1}}{\omega_i}\phi_i - \phi_{i+1} + \Delta\phi_{i(i+1)}\right), \tag{34}$$

where ϕ_i and $\phi_{(i+1)}$ are the phases of the ith and $(i+1)$th oscillators in the chain, respectively, which are defined in Eq. (27), and $\gamma_\phi > 0$ controls the convergence rate.

In the imitating mode, these phase differences are updated and will be used in Eq. (28) for the oscillator synchronization.

Shaping network. One of the most important parts of the imitating/learning agent is its *shaping network*. This part is responsible for approximating the mapping from the output of the corresponding oscillator to the desired trajectory for an actuated DoF agent. Furthermore, it is an essential block of feedback node which maps the states of the internal dynamics subspace onto its corresponding oscillator phase.

In this manuscript, a radial basis function network is used to approximate the desired mapping. By using this technique, the shaping network outputs of the agent for the ith actuated DoF are written as follows:

$$q^*_{b,i} = \sum_{k=1}^{n} w^i_k \Phi_k(\phi_i) + w^i_0, \tag{35}$$

$$\dot{q}^*_{b,i} = \sum_{k=1}^{n} v^i_k \Phi_k(\phi_i) + v^i_0, \tag{36}$$

$$u^*_i = \sum_{k=1}^{n} \theta^i_k \Phi_k(\phi_i) + \theta^i_0, \tag{37}$$

where activation function, $\Phi_k(\cdot)$, is a radial basis function (a.k.a. RBF) defined as follows:

$$\Phi_k(\phi) = \exp\left(-\frac{1}{\eta^2}(\phi - \psi_k)^2\right). \tag{38}$$

ϕ_i is the phase of the oscillator of the ith controller node, and w^i_k, v^i_k, and θ^i_k are weights of RBFs in the network.

It should be noted that, in Eq. (38), ψ_k is the center of radial basis function. These centers can be selected by various methods, e.g., randomly sampled among the input instances, obtained by Orthogonal Least Square Learning Algorithm [52] or selected via clustering of the inputs. Besides, η controls the width of RBFs which is usually considered to be constant for all RBFs.

In the learning mode, the weights of the network are adaptively changed in order to minimize the following objective function (39):

$$J_1(t) = \frac{1}{2}\left(q^*_{b,i}(t) - q_{b,i}(t)\right)^2 + \frac{1}{2}\left(\dot{q}^*_{b,i}(t) - \dot{q}_{b,i}(t)\right)^2 + \frac{1}{2}\left(u^*_i(t) - u_i(t)\right)^2. \tag{39}$$

Using the gradient descent method [53] to derive adaptive laws for the weights in the network yields the following:

$$\dot{w}^i_0 = -\gamma_w \nabla_{w^i_0} J_1 = -\gamma_w(q^*_{b,i} - q_{b,i}), \tag{40}$$

$$\dot{w}^i_k = -\gamma_w \nabla_{w^i_k} J_1 = -\gamma_w(q^*_{b,i} - q_{b,i})\Phi_k(\phi_i), \quad k \in \{1, 2, \cdots, n\}, \tag{41}$$

$$\dot{v}_0^i = -\gamma_v \nabla_{v_0^i} J_1 = -\gamma_v (\dot{q}_{b,i}^* - \dot{q}_{b,i}), \tag{42}$$

$$\dot{v}_k^i = -\gamma_v \nabla_{v_k^i} J_1, = -\gamma_v (\dot{q}_{b,i}^* - \dot{q}_{b,i}) \Phi_k(\phi_i),$$
$$k \in \{1, 2, \cdots, n\}, \tag{43}$$

$$\dot{\theta}_0^i = -\gamma_\theta \nabla_{\theta_0^i} J_1 = -\gamma_\theta (u_i^* - u_i), \tag{44}$$

$$\dot{\theta}_k^i = -\gamma_\theta \nabla_{\theta_k^i} J_1 = -\gamma_\theta (u_i^* - u_i) \Phi_k(\phi_i),$$
$$k \in \{1, 2, \cdots, n\}, \tag{45}$$

where $\nabla_x(\cdot) = \frac{\partial}{\partial x}(\cdot)$.

It should be pointed out that when the training is over and the system is in the imitating mode, the weights of the network become constant and will be used to generate the desired trajectory and control input.

In the case of internal dynamics phase mapping, the phase of internal dynamics is used as the input of a network, which generates the phase of its base frequency. In other words, the shaping network output of the imitating agent of the unactuated coordinate will be as follows:

$$\phi_0^* = \sum_{k=1}^{n} \mu_k \Phi_k(\phi_I) + \mu_0, \tag{46}$$

where $\phi_I = q_a = \pi - \theta_1$ is the input of RBF functions. The weights of RBF functions are adaptively changed by the gradient descent method in order to minimize the following objective function:

$$J_2(t) = \frac{1}{2}(\phi_0^*(t) - \phi_0(t))^2, \tag{47}$$

which yields:

$$\dot{\mu}_0 = -\gamma_\mu \nabla_{\mu_0} J_2 = -\gamma_\mu (\phi_0^* - \phi_0), \tag{48}$$

$$\dot{\mu}_k = -\gamma_\mu \nabla_{\mu_k} J_2 = -\gamma_\mu (\phi_0^* - \phi_0) \Phi_k(\phi_I),$$
$$k \in \{1, 2, \cdots, n\}, \tag{49}$$

where ϕ_0 is the oscillator output of the feedback node which is synchronized with the base frequency component of the internal dynamics in the training mode.

3.2.3. Critic agent

The assessment of the control ability of the low-level controller is made by this agent. This agent evaluates this ability using the error between the output of the shaping network and the outputs of the system. For the controller nodes, this error is defined as follows:

$$e_i = (q_{b,i} - q_{b,i}^*), \tag{50}$$

$$\dot{e}_i = (\dot{q}_{b,i} - \dot{q}_{b,i}^*), \tag{51}$$

and, for the feedback node, it is defined as follows:

$$e_0 = (\phi_0 - \phi_0^*), \tag{52}$$

$$\dot{e}_0 = (\dot{\phi}_0 - \dot{\phi}_0^*). \tag{53}$$

Then, this error will be used to generate the *assessment value* which is defined as follows:

$$u_{\beta,i} = \begin{cases} 1 & e_{\beta,i} > e_{\max} \\ 0 & \text{else,} \end{cases} \tag{54}$$

where:

$$e_{\beta,i} = k_e |e_i| + k_{\dot{e}} |\dot{e}_i|, \qquad i \in \{0, 1, \cdots, 4\}. \tag{55}$$

In other words, this value assesses the ability of the imitating agent to generate the desired trajectory within an acceptable error margin at the current time. In order to use this value and assess the ability of the imitating agent to generate the desired trajectory at all times, the history of the *assessment value* should be taken into account. In order to do that, the following dynamical system is used which is called the *assessment system*:

$$\dot{\beta}_i = -\frac{1}{\epsilon_l}(1 - u_{\beta,i})\beta_i + \epsilon_u u_{\beta,i}(1 - \beta_i), \tag{56}$$

where ϵ_l and ϵ_u are small constants, and β_i is the output of the assessment system.

This dynamical system has two different terms which have contrary effects on its output. The first one is trying to increase the output fast when the error is greater than a threshold. In other words, the error above a specific level means that the low-level control has failed to control the system. Therefore, the critic agent should act fast and assign the control task to the high-level controller. On the other hand, the second term in Eq. (56) will try to decrease the output slowly when the error drops lower than the threshold. The slow dynamics of this term helps the system to take the history of the assessment value into account.

In order to avoid Zeno phenomenon and oscillations in switching between high-level and low-level controllers and, thus, instability of the overall system, this decision is made by a switching function with hysteresis. In other words, when the system is controlled by a high-level controller and $0 < \beta_i < \beta_u$, the critic agent commands the system to be controlled by the low-level controllers. In this case, the imitating/learning agents of the low-level controllers are switched to the imitating mode. Afterwards, if $\beta_i > \beta_l > \beta_u$, the system will be switched to be controlled by the high-level controller. Therefore, imitating/learning agents of the low-level controllers will be switched back to the learning mode.

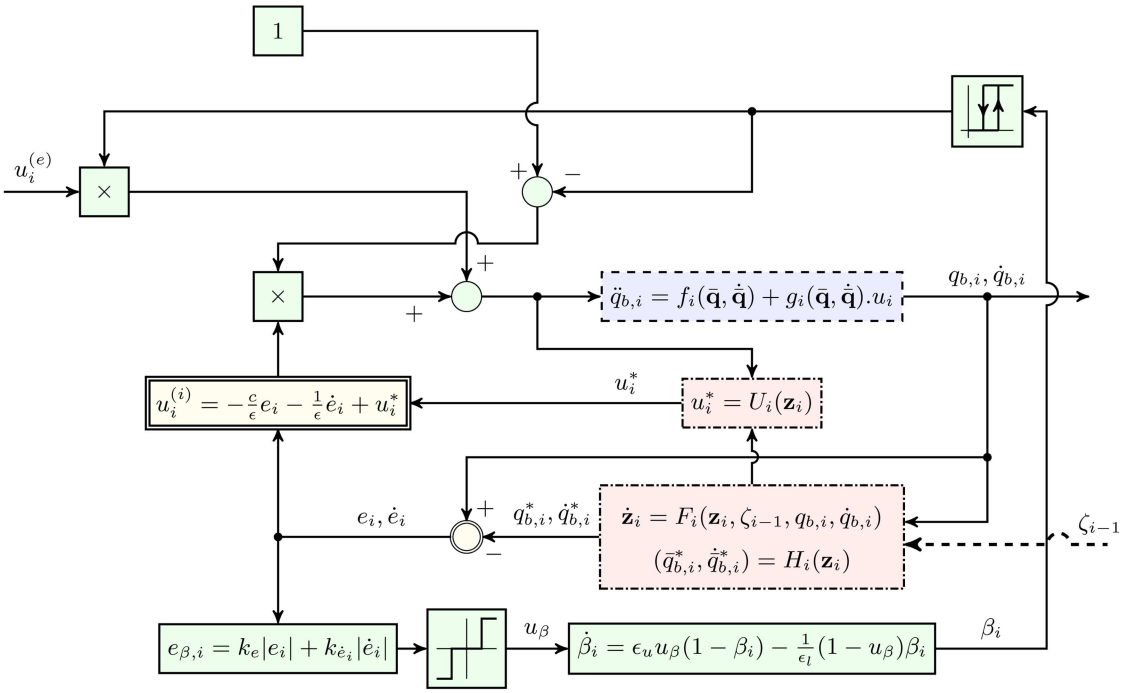

Figure 6. Implementation of the low-level controller for the ith actuated DoF. Single border blocks (green blocks in colored version) represent agent subsystems of critic; double border blocks (yellow blocks) are subsystems of control agent; dash dot border blocks (red blocks) describe the imitating/learning agent, and dash border block (blue block) is a representation of time evolution dynamics of the system. $u_i^{(e)}$ is the control action of the high-level controller and $u_i^{(i)}$ represents control action of the low-level controller for the ith actuated DoF.

3.2.4. Low-level controller in a nutshell

In brief, the structure of the low-level controller node is depicted in Figure 6. In this structure, when the system is not trained, the high-level controller's control action is directly fed into the system and its output is used to train the imitating/learning agent. When all the imitating/learning agents of the system are trained enough (the error between the output of the system and the outputs of these agents tends to zero), the critic agent cuts the high-level controller action and reroutes the input of the system to the output of the control agent.

The structure of the feedback node is just the same as that of an actuated DoF; however, it does not have a control agent and an estimator for the controller input.

4. Simulation

Consider the model in Eq. (4) with the values of the parameters for RABBIT prototype [33] presented in Table 1.

In addition to that, the parameters in Table 2 are used for constructing the virtual constraints mentioned in Eqs. (8) and (9).

It should be noted that, to stabilize the virtual constraints, feedback gains are set as $\kappa_1 = 9000$ and $\kappa_2 = 400$.

To find the approximation of the desired trajectories, the values presented in Table 3 are considered for the parameters of the imitating/learning agents.

When the system is switched to be controlled by low-level controllers, the controller agents use the values presented in Table 4 for their parameters.

In this simulation, all critic agents use the parameters values shown in Table 5.

Table 1. Mechanical parameters of RABBIT prototype (see Figure 1(a) and (b)).

Model param.	Tibia ($n=1$)	Femur ($n=2$)	Torso ($n=3$)
Mass, M_n (kg)	3.2	6.8	20
Length, l_n (m)	0.4	0.4	0.625
Mass center, ρ_n (m)	0.13	0.16	0.2
Inertia, I_n (kg m^2)	0.93	1.08	2.22

Table 2. Values of the virtual constraint parameters [32].

Output	Gain	Parameters
y_1	$k_1 = 62.5$	$\theta_{3d} = -\pi/9$ rad
y_2	$k_2 = 500$	
y_3	$k_3 = 1$	$z_{H\,\max} = 0.79$ m, $z_{H\,\min} = 0.76$ m
y_4	$k_4 = 1$	$z_{2\,\max} = 20$ mm, sld $= 0.45$ m

Table 3. The parameters values of the imitating/learning agents.

Param.	Description	Value
ε	The entrainment signal gain (see Eqs. (25) and (26))	9
γ	The gain controls the convergence rate in the governing dynamics of the oscillators (see Eq. (25))	1000
μ	Control the amplitude of oscillators (see Eq. (25))	1
k_s	The synchronization gain (see Eq. (28))	1000
γ_ϕ	The gain controls the convergence rate in Eq. (34)	800
n	Number of neurons for approximating desired trajectories(see Eqs. (35)-(37) and (46))	300
η	Width of radial basis functions (see Eq. (38))	0.1
γ_w, γ_v, γ_θ, $x\gamma_\mu$	Gradient descent gains (see Eqs. (40)-(45), (48) and (49))	10

Table 4. The parameters' values of the controller agents.

Param.	Description	Value
ϵ	The inverse of the velocity feedback gain (see Eq. (19))	1/300
c	Ratio of the position feedback gain to the velocity feedback gain (see Eq. (19))	50

Table 5. The parameters' values of the critic agents.

Param.	Description	Value
$e_{\max}$	Maximum allowable error (see Eq. (54))	0.1
k_e	Error gain (see Eq. (55))	1
$k_{\dot{e}}$	error velocity gain (see Eq. (55))	0
ϵ_l	Constant gain controls the fast dynamic time constant in Eq. (56)	0.2
ϵ_u	Constant gain controls the slow dynamic time constant in Eq. (56)	0.5
β_1	The level of the switching from the high-layer controller to the low-level controllers	0.10
β_2	The level of the switching from the low-level controllers to the high-layer controller	0.30

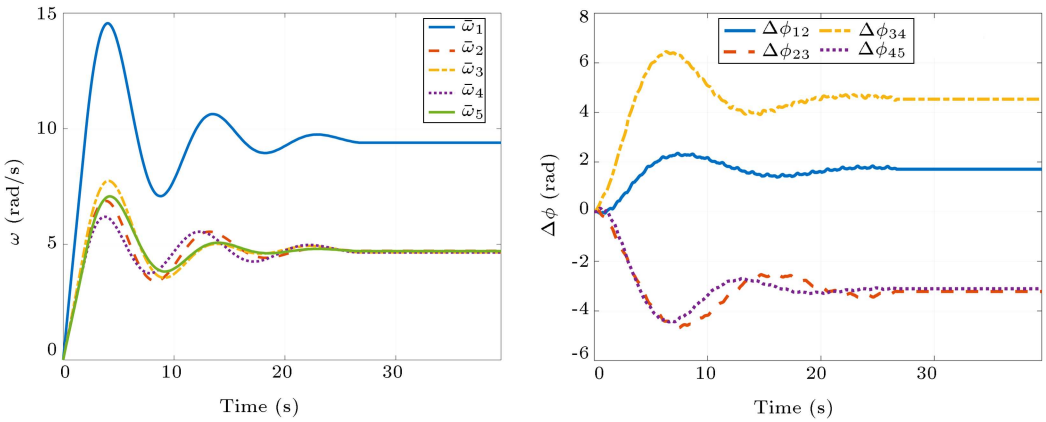

Figure 7. Time evolution of oscillators frequencies (a) and their phase differences (b) at $t = 26.7$. The system is switched to be controlled by low-level controllers. Therefore, these values do not change after the switching time.

By using the described parameters' values, the result of the simulation for more than 50 walking steps is depicted in the following figures. In Figure 7, time evolution of the frequencies of the oscillators and the phase difference between them are presented. During the training, the error between the desired trajectories (the output of the system) and the output of imitating/learning agents gets smaller and smaller until the system decides to assign control task to the low-level controllers. Figure 8 shows the position errors of the actuated DoFs. As mentioned before, when one of the legs hits the ground, the shaping networks cannot approximate the outputs accurately. Therefore, the errors increase at these times.

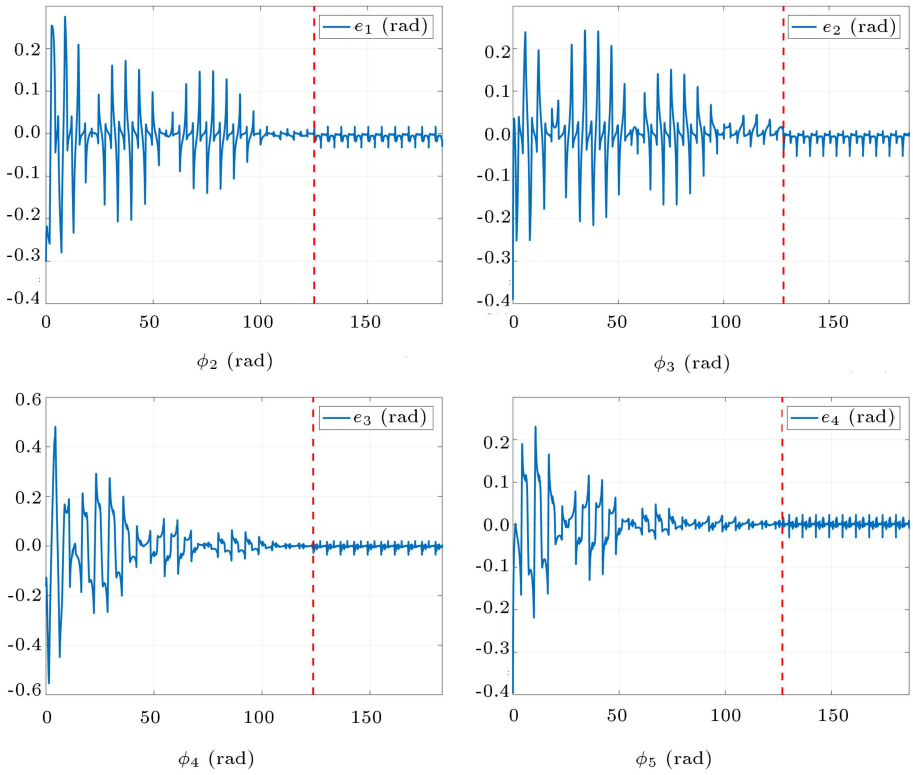

Figure 8. Error of the outputs of the imitating agents for controller nodes, i.e., $e_i = \psi_i - q^*_{b,i}$ for $i = 1 \cdots 4$. The red dashed line shows the time the system switched to be controlled by the low-level controllers network.

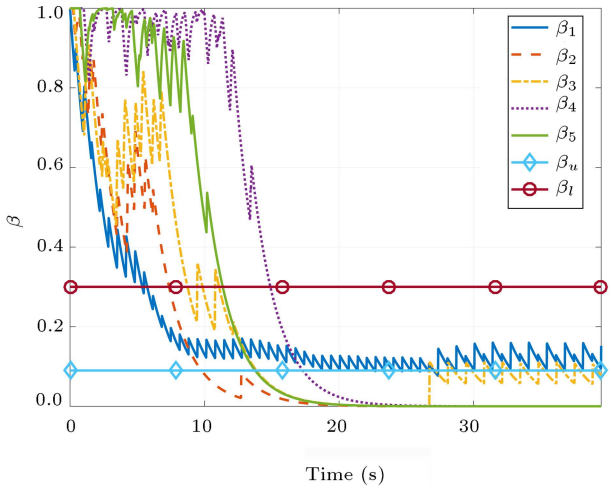

Figure 9. Outputs of the assessment systems in time.

The outputs of the critic agents are presented in Figure 9. As shown, at $t = 26.7$, the outputs of all critic agents become less than the switching value, β_l. Therefore, the system is switched to be controlled by low-level controllers at this time.

In Figure 10, the performance of the system using the high-level controller and the low-level controllers is depicted.

At last, the simulated gait is depicted in Figure 11.

5. Conclusion

In this article, the control of the walking gait for a 5-link biped robot was accomplished via a two-layer hierarchical controller.

Inspired from nature, the objective of this study was to develop a control framework for the walking gait of a biped robot to replicate the functionality of the nervous system in the animals and its learning capability under conscious trainings. To do so, a hierarchical framework was proposed which consists of two independent layers: (1) the high-level controller and (2) a network of low-level controllers.

The high-level controller was designed to replicate the function of the brain. In this study, it is a feedback linearization controller which uses a set of outputs as virtual constraints. Imposing these constraints on the system made the system achieve its desired posture. In the proposed architecture, this controller was used as a supervisory controller, which controls the system when the low-level controllers are not trained enough, or it is unable to control the system stably.

On the other hand, the low-level controllers' network was designed to replicate the functionality of the neural circuits in the spinal cord. In this design, the entities are connected together in a chain structure in which each has three components:

1. Imitating/learning agent;

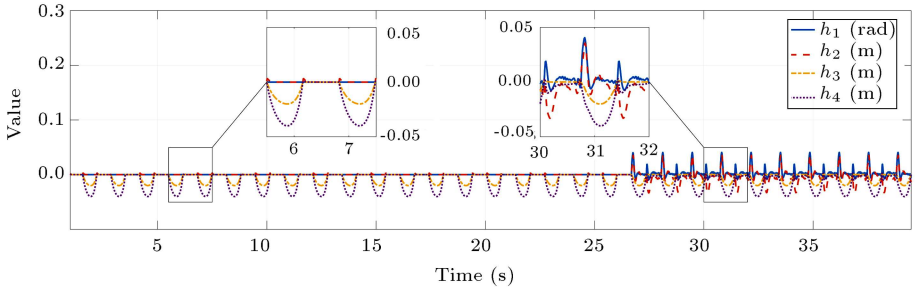

Figure 10. Time evolution of the values of the constraints defined in Eq. (6).

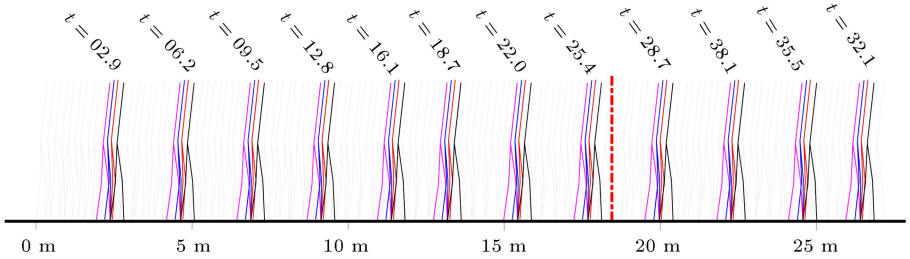

Figure 11. Simulated gait; the divider dashed red line shows the position in which the system has been switched to be controlled by the low-level controllers.

2. Critic agent;
3. Controller agent.

The imitating agent is the core of the low-level controller which learns to reproduce the desired rhythmic pattern synchronously with the other imitating agents in the network; the critic agent makes decision about switching between low-level controller network and the high-level controller network. In addition, the controller agent makes the system follow a desired pattern generated by the imitating agent.

In a nutshell, the proposed method works as follows: First, the system is controlled by the high-level controller. Meanwhile, the low-level controllers are trained and their outputs are evaluated at the same time. Next, if all low-level controllers in the network have the ability to generate the desired trajectories, the high-level controller gets turned off and the system will be controlled by low-level controllers. The result shows the applicability of this method in the walking gait of biped robots. Therefore, the developed framework controls an unstable under-actuated biped robot using simple identical controllers which have no prior knowledge about the system. This makes the control of the system modular, simple, and computationally efficient.

References

1. Mohammadi, E., Zohoor, H., and Khadem, S. "Design and prototype of an active assistive exoskeletal robot for rehabilitation of elbow and wrist", *Scientia Iranica*, **23**(3), pp. 998-1005 (2016).
2. Gritli, H. and Belghith, S. "Walking dynamics of the passive compass-gait model under ogy-based state-feedback control: Analysis of local bifurcations via the hybrid poincaré map", *Chaos, Solitons and Fractals*, **98**, pp. 72-87 (2017).
3. Razavi, H., Bloch, A., Chevallereau, C., and Grizzle, J. "Symmetry in legged locomotion: a new method for designing stable periodic gaits", *Autonomous Robots*, **41**(5), pp. 1119-1142 (2017).
4. Zhao, H., Hereid, A., Ma, W.-L., and Ames, A. "Multi-contact bipedal robotic locomotion", *Robotica*, **35**(5), pp. 1072-1106 (2017).
5. Gupta, S. and Kumar, A. "A brief review of dynamics and control of underactuated biped robots", *Advanced Robotics*, pp. 1-17 (2017).
6. Luo, J.-W., Fu, Y.-L., and Wang, S.-G. "3d stable biped walking control and implementation on real robot", *Advanced Robotics*, pp. 1-16 (2017).
7. Hasankola, M., Ehsaniseresht, A., Moghaddam, M., and Mirzaei Saba, A. "Analysis, modeling, manufacturing and control of an elastic actuator for rehabilitation robots", *Scientia Iranica*, **22**(5), pp. 1855-1865 (2015).
8. Ataei, M., Salarieh, H., and Alasty, A. "An adaptive impedance control algorithm; application in exoskeleton robot", *Scientia Iranica*, **22**(2), pp. 519-529 (2015).
9. Kulic, D., Venture, G., Yamane, K., Demircan, E., Mizuuchi, I., and Mombaur, K. "Anthropomorphic movement analysis and synthesis: A survey of methods and applications", *IEEE Transactions on Robotics*, **32**(4), pp. 776-795 (2016).
10. Hurmuzlu, Y., Génot, F., and Brogliato, B. "Modeling, stability and control of biped robots-a general framework", *Automatica*, **40**(10), pp. 1647-1664 (2004).

11. Grizzle, J.W., Chevallereau, C., Sinnet, R.W., and Ames, A.D. "Models, feedback control, and open problems of 3D bipedal robotic walking", *Automatica*, **50**(8), pp. 1955-1988 (2014).
12. Luksch, T. "Human-like control of dynamically walking bipedal robots", PhD thesis, University of Kaiserslautern, Kaiserslautern (2010).
13. Bora, N.M., Molke, G.V., and Munot, H.R. "Understanding human gait: A survey of traits for biometrics and biomedical applications", *2015 International Conference on Energy Systems and Applications*, pp. 723-728 (2015).
14. Beigzadeh, B. and Meghdari, A. "On dynamic nonprehensile manipulation of multibody objects", *Scientia Iranica, Transactions B, Mechanical Engineering*, **22**(2), pp. 467-486 (2015).
15. Rossignol, S. "Dynamic sensorimotor interactions in locomotion", *Physiological Reviews*, **86**(1), pp. 89-154 (2006).
16. Dietz, V. "Spinal cord pattern generators for locomotion", *Clinical Neurophysiology*, **114**(8), pp. 1379-1389 (2003).
17. Guertin, P.A. "The mammalian central pattern generator for locomotion", *Brain Research Reviews*, **62**(1), pp. 45-56 (2009).
18. Hooper, S.L. "Central pattern generators", *Encyclopedia of Life Sciences*, John Wiley & Sons, Ltd, pp. 1-9 (2001).
19. Villarreal, D.J. and Gregg, R.D. "A survey of phase variable candidates of human locomotion", *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, **2014**, pp. 4017-4021 (2014).
20. Wang, X., Ding, F., Alsaadi, F.E., and Hayat, T. "Convergence analysis of the hierarchical least squares algorithm for bilinear-in-parameter systems", *Circuits Syst. Signal Process*, **35**(12), pp. 4307-4330 (2016).
21. Ma, J., Xiong, W., Chen, J., and Ding, F. "Hierarchical identification for multivariate Hammerstein systems by using the modified Kalman filter", *IET Control Theory Applications*, **11**(6), pp. 857-869 (2017).
22. Wang, D., Mao, L., and Ding, F. "Recasted models-based hierarchical extended stochastic gradient method for MIMO nonlinear systems", *IET Control Theory Applications*, **11**(4), pp. 476-485 (2017).
23. Ding, F. and Wang, X. "Hierarchical stochastic gradient algorithm and its performance analysis for a class of bilinear-in-parameter systems", *Circuits, Systems, and Signal Processing*, **36**(4), pp. 1393-1405 (2017).
24. Wang, X. and Ding, F. "Recursive parameter and state estimation for an input nonlinear state space system using the hierarchical identification principle", *Signal Processing*, **117**, pp. 208-218 (2015).
25. Wang, Y. and Ding, F. "The auxiliary model based hierarchical gradient algorithms and convergence analysis using the filtering technique", *Signal Processing*, **128**, pp. 212-221 (2016).
26. Wang, Y. and Ding, F. "Novel data filtering based parameter identification for multiple-input multiple-output systems using the auxiliary model", *Automatica*, **71**, pp. 308-313 (2016).
27. Chen, H., Xiao, Y., and Ding, F. "Hierarchical gradient parameter estimation algorithm for hammerstein nonlinear systems using the key term separation principle", *Appl. Math. Comput.*, **247**(C), pp. 1202-1210 (2014).
28. Ding, F., Liu, X., Chen, H., and Yao, G. "Hierarchical gradient based and hierarchical least squares based iterative parameter identification for CARARMA systems", *Signal Processing*, **97**, pp. 31-39 (2014).
29. Ma, J., Yuan, L., Zhao, Z., and He, F. "Transmission loss optimization-based optimal power flow strategy by hierarchical control for dc microgrids", *IEEE Transactions on Power Electronics*, **32**(3), pp. 1952-1963 (2017).
30. Zhao, J., Wong, P.K., Ma, X., and Xie, Z. "Chassis integrated control for active suspension, active front steering and direct yaw moment systems using hierarchical strategy", *Vehicle System Dynamics*, **55**(1), pp. 72-103 (2017).
31. Grizzle, J.W., Abba, G., and Plestan, F. "Asymptotically stable walking for biped robots: analysis via systems with impulse effects", *IEEE Transactions on Automatic Control*, **46**(1), pp. 51-64 (2001).
32. Plestan, F., Grizzle, J.W., Westervelt, E.R., and Abba, G. "Stable walking of a 7-DOF biped robot", *IEEE Transactions on Robotics and Automation*, **19**(4), pp. 653-668 (2003).
33. Westervelt, E.R., Grizzle, J.W., and Koditschek, D.E. "Hybrid zero dynamics of planar biped walkers", *IEEE Transactions on Automatic Control*, **48**(1), pp. 42-56 (2003).
34. Hurmuzlu, Y. "Dynamics of bipedal gait Part II-Stability analysis of a planar five-link biped", *Journal of Applied Mechanics*, **60**(2), pp. 337-343 (1993).
35. Djoudi, D., Chevallereau, C., and Aoustin, Y. "Optimal reference motions for walking of a biped robot", *2005 IEEE International Conference on Robotics and Automation*, pp. 2002-2007 (2005).
36. de Pina Filho, A.C., Dutra, M.S., and Santos, L. "Modelling of bipedal robots using coupled nonlinear oscillators", *Mobile Robots Towards New Applications*, A. Lazinica, Ed., InTech, Ch. 4, pp. 55-78 (2006).
37. Mondal, S., Nandy, A., Chandrapal, Chakraborty, P., and Nandi, G.C. "A central pattern generator based nonlinear controller to simulate biped locomotion with a stable human gait oscillation", *International Journal of Robotics and Automation (IJRA)*, **2**(2), pp. 77-127 (2011).
38. Liu, G.L., Habib, M.K., Watanabe, K., and Izumi, K. "Central pattern generators based on Matsuoka oscillators for the locomotion of biped robots", *Artificial Life and Robotics*, **12**(1-2), pp. 264-269 (2008).

39. Aoi, S. and Tsuchiya, K. "Locomotion control of a biped robot using nonlinear oscillators", *Autonomous Robots*, **19**(3), pp. 219-232 (2005).
40. Nassour, J., Hénaff, P., Benouezdou, F., and Cheng, G. "Multi-layered multi-pattern CPG for adaptive locomotion of humanoid robots", *Biological Cybernetics*, **108**(3), pp. 291-303 (2014).
41. Liu, C., Yang, J., Bu, W., and Chen, Q. "A trajectory generation method for biped walking based on neural oscillators", *13th International Conference on Networking, Sensing, and Control (ICNSC)*, pp. 1-6 (2016).
42. Yu, J., Tan, M., Chen, J., and Zhang, J. "A survey on CPG-inspired control models and system implementation", *IEEE Transactions on Neural Networks and Learning Systems*, **25**(3), pp. 441-456 (2014).
43. Spong, M.W. "Partial feedback linearization of underactuated mechanical systems", *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94)*, pp. 314-321 (1994).
44. Buche, G., Chevallereau, C., Abba, G., Aoustin, Y., Plestan, F., Westervelt, E.R., Canudas-de-Wit, C., and Grizzle J.W. "RABBIT: A testbed for advanced control theory", *IEEE Control Systems Magazine*, **23**(5), pp. 57-79 (2003).
45. Westervelt, E.R. and Grizzle, J.W. "Design of asymptotically stable walking for a 5-link planar biped walker via optimization", *IEEE International Conference on Robotics and Automation*, pp. 3117-3122 (2002).
46. Liu, C. and Su, J. "Biped Walking control using offline and online optimization", *30th Chinese Control Conference (CCC)*, pp. 3472-3477 (2011).
47. Verhulst, F. "Methods and applications of singular perturbations", *Boundary Layers and Multiple Timescale Dynamics*, **50**, Springer Science & Business Media, New York, NY (2005).
48. Prentice, S.D., Patla, A.E., and Stacey, D.A. "Simple artificial neural network models can generate basic muscle activity patterns for human locomotion at different speeds", *Experimental Brain Research*, **123**(4), pp. 474-480 (1998).
49. Righetti, L., Buchli, J., and Ijspeert, A.J. "Dynamic Hebbian learning in adaptive frequency oscillators", *Physica D: Nonlinear Phenomena*, **216**(2), pp. 269-281 (2006).
50. Wang, W. and Slotine, J.-J.E. "On partial contraction analysis for coupled nonlinear oscillators", *Biological Cybernetics*, **92**(1), pp. 38-53 (2004).
51. Righetti, L. and Ijspeert, A.J. "Programmable central pattern generators: an application to biped locomotion control", *IEEE International Conference on Robotics and Automation*, pp. 1585-1590 (2006).
52. Chen, S., Cowan, C.F.N., and Grant, P.M. "Orthogonal least squares learning algorithm for radial basis function networks", *IEEE Transactions on Neural Networks*, **2**(2), pp. 302-309 (1991).
53. Saad, D., *On-Line Learning in Neural Networks*, Cambridge University Press (2009).
54. Khalil, H.K., *Nonlinear Systems*, Pearson Education Prentice Hall (2002).

Appendix A

Definition of Parameters in the perturbed dynamics

In this section, the variables used in Eq. (18) are given by:

$$\begin{cases} \Delta_i^{(1)} = \bar{C}_{ii}\delta\dot{\bar{q}}_j + \sum\limits_k \left[\frac{\partial \bar{D}_{ik}}{\partial \bar{q}_i}\ddot{\bar{q}}_k^* + \frac{\partial \bar{C}_{ik}}{\partial \bar{q}_i}\dot{\bar{q}}_k^*\right] \\ \Delta_i^{(2)} = \frac{\partial \bar{g}_i}{\partial \bar{q}_i} + \sum\limits_k \left[\frac{\partial \bar{D}_{ik}}{\partial \bar{q}_i}\ddot{\bar{q}}_k^* + \frac{\partial \bar{C}_{ik}}{\partial \bar{q}_i}\dot{\bar{q}}_k^*\right] \\ \Delta_i^{(3)} = \sum\limits_{j\neq i}\Bigg(\bar{D}_{ij}\delta\ddot{\bar{q}}_j + \delta_j\bar{g}_i + \bar{C}_{ij}\delta\dot{\bar{q}}_j + \\ \qquad \sum\limits_k \left[\delta_j\bar{D}_{ik}\ddot{\bar{q}}_k^* + \delta_j\bar{C}_{ik}\dot{\bar{q}}_k^*\right]\Bigg) \end{cases} \tag{A.1}$$

where C_{ij} and D_{ij} denote row i and column j of matrix $\mathbf{C}$ and matrix $\mathbf{D}$, respectively. In addition, g_i denotes the ith component of vector $\mathbf{g}$. Moreover:

$$\delta_j = \delta\bar{q}_j\frac{\partial}{\partial\bar{q}_j} + \delta\dot{\bar{q}}_j\frac{\partial}{\partial\dot{\bar{q}}_j}. \tag{A.2}$$

Appendix B

Perturbation analysis of dynamics of an actuated DoF with the high-gain controller

In this section, singular perturbation analysis of Eq. (20) is considered.

Without loss of generality, let us assume that there exists a regular expansion for the solution, i.e.:

$$\begin{aligned} \delta\bar{q}_i(t) =& \delta Q_i^{(0)}(t/\epsilon) + \epsilon\delta Q_i^{(1)}(t/\epsilon) + \cdots \\ &+ \delta q_i^{(0)}(t) + \epsilon\delta q_i^{(1)}(t) + \cdots . \end{aligned} \tag{B.1}$$

It should be noted that, due to the singularity of the system, $Q_i^{(n)}(t/\epsilon)$ for $n = 0, 1, \cdots$ is added to the solution to prevent the boundary layer jump near $t = 0$. Substituting the proposed expansion into Eq. (20) yields:

$$\begin{cases} \delta\dot{q}_i^{(0)} + c\delta q_i^{(0)} = 0, \\ \delta\dot{q}_i^{(1)} + c\delta q_i^{(1)} = -\Delta_i^{(3)} - \bar{D}_{ii}\delta\ddot{q}_i^{(0)} \\ \qquad -\Delta_i^{(1)}\delta\dot{q}_i^{(0)} - \Delta_i^{(2)}\delta q_i^{(0)} \\ \delta\dot{q}_i^{(n)} + c\delta q_i^{(n)} = -\bar{D}_{ii}\delta\ddot{q}_i^{(n-1)} - \Delta_i^{(1)}\delta\dot{q}_i^{(n-1)} \\ \qquad -\Delta_i^{(2)}\delta q_i^{(n-1)}, \qquad n \in \{2, 3, \cdots\}. \end{cases} \tag{B2.}$$

Therefore, the solution of these terms at the lowest order is written as follows:

$$\begin{cases} \delta q_i^{(0)}(t) = Ae^{-ct} \\ \delta \dot{q}_i^{(0)}(t) = -cAe^{-ct} \end{cases} \tag{B.3}$$

where A is a constant determined by the initial conditions.

For the boundary layer solution, the proposed expansion yields the following equations:

$$\begin{cases} \bar{D}_{ii}\delta\ddot{Q}_i^{(0)} + \delta\dot{Q}_i^{(0)} = 0, \\ \bar{D}_{ii}\delta\ddot{Q}_i^{(1)} + \delta\dot{Q}_i^{(1)} = -c\delta Q_i^{(0)} - \Delta_i^{(1)}\delta\dot{Q}_i^{(0)}, \\ \bar{D}_{ii}\delta\ddot{Q}_i^{(2)} + \delta\dot{Q}_i^{(2)} = -c\delta Q_i^{(1)} - \Delta_i^{(1)}\delta\dot{Q}_i^{(1)} \\ \qquad\qquad - \Delta_i^{(2)}Q_i^{(0)} - \Delta_i^{(3)}, \\ \bar{D}_{ii}\delta\ddot{Q}_i^{(n)} + \delta\dot{Q}_i^{(n)} = -c\delta Q_i^{(n-1)} - \Delta_i^{(n-1)}\delta\dot{Q}_i^{(n-1)} \\ \qquad\qquad - \Delta_i^{(2)}Q_i^{(n-2)}, \qquad n \in \{3, 4, \cdots\}. \end{cases} \tag{B.4}$$

In these equations, $0 < \bar{D}_{ii} < \infty$ ($\bar{\mathbf{D}}(t)$ is a positive definite matrix for $\forall t \in \mathbb{R}$); therefore, the lowest order term of the boundary layer solution is exponentially stable [54, p. 154] and decays with rate t/ϵ:

$$\begin{cases} \delta\dot{Q}_i^{(0)}(t/\epsilon) = B\delta_D(t/\epsilon) \\ \delta Q_i^{(0)}(t/\epsilon) = \epsilon B\int_0^{t/\epsilon}\delta_D(\tau)d\tau \end{cases} \tag{B.5}$$

where $\delta_D(\cdot)$ is an exponentially decaying function with the rate of its argument and $\delta_D(0) = 1$, and B is a constant determined by the initial conditions.

Therefore, the solution of the system will be as follows:

$$\begin{cases} \delta\bar{q}_i(t) = Ae^{-ct} + \mathcal{O}(\epsilon), \\ \delta\dot{\bar{q}}_i(t) = -cAe^{-ct} + B\delta_D(t/\epsilon) + \mathcal{O}(\epsilon) \end{cases} \tag{B.6}$$

Hence, satisfying the initial conditions yields:

$$\begin{cases} A = \delta\bar{q}_i(0), \\ B = \delta\dot{\bar{q}}_i(0) + c\delta q_i(0). \end{cases} \tag{B.7}$$

Appendix C

Proof of Theorem 2

Clearly, if system (22) could be expressed in the form of Eqs. (23) and (24), the theorem is proved and the mentioned transformation exists.

It is obvious that the local dynamical behavior of a system with n states can be fully expressed by a set of its projections on n_1 linear independent subspaces U_i where $n_1 \leq n$ and $\sum_{i=1}^{n_1}\dim(U_i) = n$. Now, consider the projection of the system in Eq. (22) on subspace U_i where $\dim(U_i) = d_i$. By taking the assumption into consideration, the system in Eq. (22) has an oscillatory stable trajectory with the period of T. It means that the trajectory of the system is a closed orbit in $\mathbb{R}^n$ and its projection on any subspace. Therefore, the projected dynamics of the system on U_i is a stable closed trajectory, and its states can be divided into a monotonic state and $(d_i - 1)$ asymptotically stable ones. The monotonic state in this subsystem can be interpreted as the phase of the oscillation. Due to the oscillatory nature of the system, it can be also concluded that its time derivative is a periodic function with the period of T with respect to this state. Besides, by considering the assumption that the trajectory of the system is an asymptotically stable one, it can be concluded that any deviation from the trajectory will be forgotten in subspace U_i and the system will asymptotically converge to the trajectory. This means that there is an asymptotically stable equilibrium point for the other $(d_i - 1)$ states of the subspace U_i and without loss of generality, it can be assumed that it is located at the origin. Hence, the dynamics of the system on subspace U_i is governed by the following equations:

$$\dot{\xi}_i = A_i(\xi, \eta), \tag{C.1}$$

$$\dot{\eta}_i = \mathbf{B}_i(\xi, \eta), \tag{C.2}$$

where ξ_i is monotonic on the trajectory projected into subspace U_i, and η_i denotes asymptotically stable states of subspace U_i. It should be noted that $\xi = (\xi_1, \xi_2, \cdots, \xi_{n_1}) \in \mathbb{R}^{n_1}$ is the stacked vector of monotonic states of all aforesaid subspaces, and $\eta = (\eta_1, \eta_2, \cdots, \eta_{n_1}) \in \mathbb{R}^{n-n_1}$ is the stacked vector of asymptotically stable states of the subspaces.

Appendix D

Contraction analysis of driven Hopf oscillator

Consider the driven Hopf oscillator system:

$$\begin{cases} \dot{z}_1 = \gamma\left(\mu^2 - k_s/\gamma - z_1^2 - z_2^2\right)z_1 - \omega z_2 + u_1 \\ \dot{z}_2 = \gamma\left(\mu^2 - k_s/\gamma - z_1^2 - z_2^2\right)z_2 + \omega z_1 + u_2. \end{cases} \tag{D.1}$$

The corresponding Jacobian matrix can be written as follows:

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 & \mathbf{J}_2 \end{bmatrix}, \tag{D.2}$$

where:

$$\mathbf{J}_1 = \begin{bmatrix} -\gamma\left(3z_1^2 + z_2^2 + k_s/\gamma - \mu^2\right) \\ \omega - 2\gamma z_1 z_2 \end{bmatrix}, \tag{D.3}$$

and:

$$\mathbf{J}_2 = \begin{bmatrix} -\omega - 2\gamma z_1 z_2 \\ -\gamma\left(3z_2^2 + z_1^2 + k_s/\gamma - \mu^2\right) \end{bmatrix}. \tag{D.4}$$

This matrix is negative definite for $k_s > \gamma\mu^2$. Therefore:

$$\frac{d}{dt}\left(\delta\mathbf{z}^{\mathbf{T}}\delta\mathbf{z}\right) = 2\delta\mathbf{z}^{\mathbf{T}}\mathbf{J}\delta\mathbf{z} < 0, \tag{D.5}$$

where $\delta\mathbf{z}$ is a virtual displacement between two neighboring solution trajectories. It implies that the system is contracting for $k_s > \gamma\mu^2$ [50].

Biographies

Masoud Yazdani received his BS and MSc degrees in Mechanical Engineering from Sharif University, Tehran, Iran in 2008 and 2010, respectively. He is currently pursuing his PhD degree at the Mechanical Engineering Department, Sharif University of Technology, Tehran, Iran. His research interests include development, modeling and control of bio-inspired robots, especially control of legged robots.

Hassan Salarieh received his BSc degree in Mechanical Engineering and also Pure Mathematics from Sharif University of Technology, Tehran, Iran in 2002. He graduated from the same university with MSc and PhD degrees in the same field of study in 2004 and 2008. At present, he is a Professor at Sharif University of Technology. His fields of research are dynamical systems, control theory, and stochastic systems.

Mahmood Saadat Foumani received his PhD degree in Mechanical Engineering from Sharif University of Technology, Tehran, I.R. Iran in 2002. He was a Faculty member at Semnan University from 2002 to 2006 and is now a faculty member in Sharif University of Technology, Mechanical Engineering Department. He teaches courses at the Applied Design group at undergraduate and graduate levels. His teaching focuses on mechanical Engineering design, vehicle dynamics, chassis design, and advanced mathematics.