International Journal of Control, Automation and Systems 16(6) (2018) 2935-2947
http://dx.doi.org/10.1007/s12555-017-0578-0 ISSN:1598-6446 eISSN:2005-4092
http://www.springer.com/12555

Bio-inspired Decentralized Architecture for Walking of a 5-link Biped Robot with Compliant Knee Joints

Masoud Yazdani, Hassan Salarieh*, and Mahmoud Saadat Foumani

Abstract: Animal walking is one of the most robust and adaptive locomotion mechanisms in the nature, involves sophisticated interactions between neural and biomechanical levels. It has been suggested that the coordination of this process is done in a hierarchy of levels. The lower layer contains autonomous interactions between muscles and spinal cord and the higher layer (e.g. the brain cortex) interferes when needed. Inspiringly, in this study we present a hierarchical control architecture with a state of the art intrinsic online learning mechanism for a dynamically walking 5-link biped robot with compliant knee joints. As the biological counterpart, the system is controlled by independent control units for each joint at the lower layer. In order to stabilize the system, these units are driven by a sensory feedback from the posture of the robot. A central stabilizing controller at the upper layer arises in case of failing the units to stabilize the system. Consequently, the units adapt themselves by including online learning mechanism. We show that using this architecture, a highly unstable system can be stabilized with identical simple controller units even though they do not have any feedback from all other units of the robot. Moreover, this architecture may help to better understand the complex motor tasks in human.

Keywords: Biped walking, dcentralized control, dynamic robot, hierarchical control, legged locomotion, online learning.

1. INTRODUCTION

Under-actuated robots and especially biped robots have been a research hotspot in the last decades and gave rise to fascinating robots and products. To achieve such great products, the wide range of topics from studying biological locomotion and their mechanical model [1], model formulation [1, 2], methods of gait synthesis [2] and the mechanical realization of biped robots [3] to the control of such systems [4, 5] have been addressed by the literature. For example in [6] dynamic walking of biped robots and in [7] human-gait analysis for biomedical applications have been studied. Optimal path planning for biped robot running has been studied in [8], and robust predictive control with on-line gait generation for biped robots has been investigated in [9].

A biped robot like a multi-link inverted pendulum is intrinsically unstable. In addition, for walking, the dynamics of the system is continuously switched between some conditions, and we have a hybrid nonlinear system which should present a cyclic motion. So locomotion is a very complicated control task, and the control system must use the instantaneous values of many state variables and generate the output values of many actuators for stabilizing the system and providing a cyclic motion called walking. Recent studies show that locomotion control in mammals, is based on the neural circuit activities within the spinal cord which is usually called CPG. It seems that the mentioned circuits are responsible for learning rhythmic activities. In other words, the locomotion is initiated by the brain and the mentioned neural circuits learn the pattern. Then, the neural circuits control the muscle activation that makes locomotion possible.

This idea is discussed and developed in the latest decades. Cronin *et al.* in [10] developed a fully actuated biped robot with a two layer controller. In this study, the trajectory of each joint is generated from human gait. Also, Odashima *et al.* in [11] present a two layer controller architecture for a hexapod robot where the upper layer controller is responsible for generating the desired trajectory and the lower layer is making sure that the robot follows that path. This idea also implemented in [12, 13]. However, in all these studies, the lower layer controller and the upper layer one are two pieces of a controller that without each ones, the system does not perform. Also, these architectures cannot be implemented on under-actuated and unstable systems.

Therefore, the main objective of this article is to develop a two-layer controller framework for a walking biped which replicates the function of brain in the learning

Manuscript received September 21, 2017; revised May 18, 2018; accepted July 9, 2018. Recommended by Associate Editor Kang-Hyun Jo under the direction of Editor Hyun-Seok Yang.

Masoud Yazdani, Hassan Salarieh, and Mahmood Saadat Foumani are with the School of Mechanical Engineering, Sharif University of Technology, Tehran, Iran (e-mails: masoudyazdani@mech.sharif.ir, {salarieh, m_saadat}@sharif.edu).
* Corresponding author.

process at the upper layer and the function of the neural circuits in the control of locomotion at the lower layer. It means that we have a two-level independent control system. In upper layer, similar to the brain function, a high performance control system produces the locomotion by using the whole state variables and providing the outputs of all actuators. In the lower layer, which is also called low-level control, the system learns to produce controlling signals of actuators to have a stable walking pattern without any need to the whole state variables of the system. Indeed, in the second layer we have a distributed low-level controls with learning capability. In this architecture, before the controllers in the lower layer (low-level controllers) get tuned, the system is controlled by a high-level controller at the upper layer which has a detailed knowledge of the system dynamics. Then, when the parameters of the lower level controllers get tuned, control of the system is entrusted to them. In this structure, due to their poor knowledge of the system dynamics, low-level controllers cannot be expected to control the system robustly under moderate to heavy disturbances. In these circumstances, it would be desired to put the overall system control in the hand of upper layer controller as a supervisory control in order to bring the system's output back on track.

In the high-level controller of the proposed framework a dynamic-based strategy [14–17] is utilized. In this strategy, the main focus is on the dynamics and kinematics of the system to generate gait motion, and a manifold of constraints between the state variables of the system is considered to be asymptotically stabilized by the controller. Stabilizing the mentioned constraints results in dynamic walking. Although such an approach needs extensive knowledge of the mechanical structure and high computational power, it performs more robustly than trajectory-based controllers under moderate disturbances.

The low-level controllers in this framework are a network of simple trajectory based controllers which are working based on the pre-calculated joint trajectories. In this approach, the reference trajectories are pre-computed via various methods and then these trajectories are pursued by means of feedback controllers. To compute these trajectories, some used optimization of various cost functions over a walking cycle [8, 18, 19]; and some extracted them from their analogy with biological or simpler mechanical systems [20, 21]. In this paper the trainable neural network of the low-level control uses the controlled trajectories and control signals of the high-level control to produce the reference trajectory and feed-forward part of the low-level controller. Also each low-level control uses a local feedback of the neighboring links, and also a common variable shown the posture of the robot. So the low-level controllers have simple structures with local feedbacks and use a common feedback depicted the posture of the robot. Also they utilize a trainable neural network which can be trained to produce reference trajectories of the joints. Indeed, the proposed framework combines the approach of trajectory-based and dynamic based methods. Thus, it takes advantage of the simplicity and computational efficiency of trajectory-based approach and the robustness of dynamic-based approach at the same time. The proposed method utilizes the hierarchy concept of the brain-neural circuits of mammals and try to reproduce such concept for controlling the biped robots, and depict for the first time, a simple structured distributed control network for locomotion. This framework can mimic the action of nature in controlling complicated hybrid systems with a network of simple controllers.

In Section 2, the dynamical model of a five-link biped robot with passive knee joint is driven. Then, the controller structure for this robot is proposed in Section 3. 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 conclusion is drawn in Section 5.

2. ROBOT MODEL

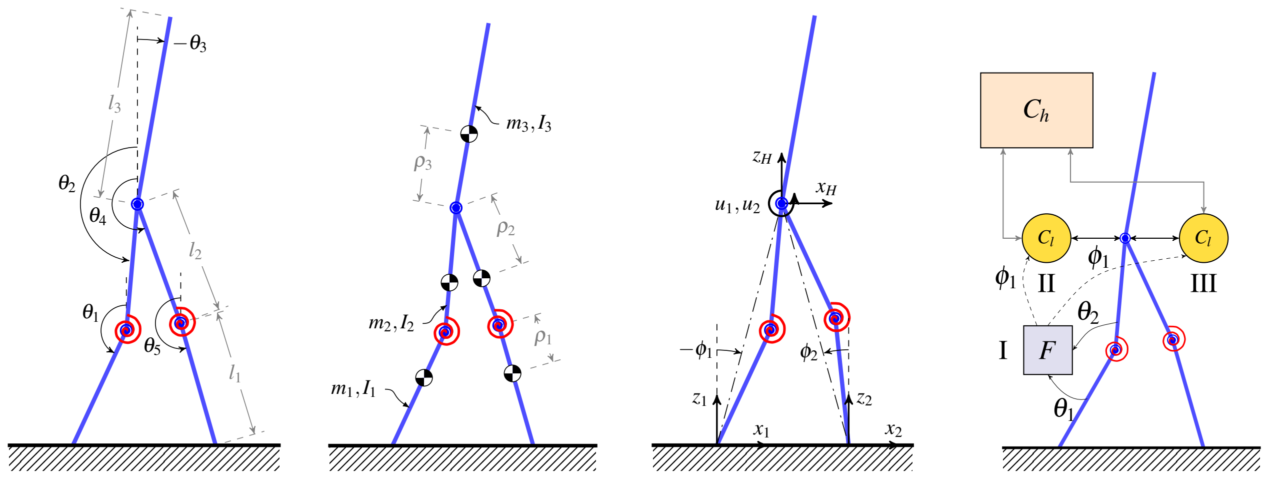
The model has a torso link and two identical kinematic open chains represent the legs which are pivoted together at a point called hip (Fig. 1(a)). Each leg consists of two links hinged together at knee. Knee joints are considered as passive joints modeled as a pair of spring and damper and hip joints are considered as active joints which are torque exerting actuators used as the inputs of the system.

It is assumed that the motion is confined to sagittal plane and the walking of the robot is considered on a level surface. Beside these assumption, the walking of the robot is interpreted as consecutive single support phases (meaning only one leg is on the ground and act as a pivot) and transition from one leg to another one is taking place in an infinitesimal length of time [14]. Therefore, the model of the robot can be described by two parts: (1) A differential equation representing the governing dynamics during single support phases; and (2) An impulse model representing the contact event (modeled as a contact between rigid bodies).

2.1. Single support phase model

During the single support phase, the stance leg is acting as a pivot. Therefore, the dynamic model of the robot during this phase has five DoFs. Let $\mathbf{q} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)^{\mathrm{T}}$ be the set of coordinates depicted in Fig. 1(a); and $\mathbf{u} = (u_1, u_2)^{\mathrm{T}}$ is the inputs of the system presented in Fig. 1(c). Since only symmetric gaits are of interest, the same model can be used irrespective of which leg is the stance leg if the coordinates are relabeled after each impact [22]. By using the Lagrange method, the equation of motion for the robot can be derived as the following equation,

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{B}(\mathbf{q})\mathbf{u} - \kappa\mathbf{q} - \beta\dot{\mathbf{q}}. \quad (1)$$



(a) Schematic of biped robot showing all the absolute angles (θ_i) and the lengths of the links (l_i).

(b) schematic of biped robot showing all the masses (m_i); moments of inertias (I_i); and the locations of center of masses (ρ_i).

(c) Schematic of biped robot showing all the inputs (u_i). ϕ_1 and ϕ_2 are the hip posture angle with respect to the stance leg end and swing end leg, respectively; (x_1, z_1), (x_2, z_2) and (x_H, z_H) are location of stance leg, swing leg and the hip, respectively.

(d) Schematic of low-level controllers network (C_l and F) and their interaction with the high-level controller (C_h).

Fig. 1. Schematic of the biped robot.

The matrix **D** is the inertia matrix; **C** is the Coriolis matrix; **g** is the gravity vector and κ and β are the spring and damping constant matrices of passive joints. Furthermore, the matrix **B** maps inputs of the system to the generalized forces. This equation is expressed in state-space form for better readability as;

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

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

2.2. Impact model

The impact between the swing leg and the ground is modeled as a contact between two rigid bodies. At the contact event, the following conditions are assumed [23].

1) The impact is instantaneous. The impulsive forces of the impact result in an instantaneous change in the velocities, but not instantaneous change in the angles and positions.
2) The contact of the swing leg with the ground results in no rebound and no slipping of the swing leg and the stance leg lifts from the ground without interaction.

These hypotheses yield an expression for the links' velocities after the impact in terms of the velocities just before the impact; e.g. the impact model results in a smooth discrete map [15],

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

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, the function Δ is responsible for the mapping from the states of the system before the impact to the states of the system after the impact. Moreover, it reorders the states (i.e., relabeling the swing leg and the stance leg) in order to be used as the initial conditions for the next swing phase equation of motion.

2.3. Overall model

The overall dynamic system can be expressed as a hybrid system [15],

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

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

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

In other words, the trajectory evolution of the system is described by (2) until the impact occurs (when the states of the system belong to $\mathcal{S}$). The impact, which is described by (3), changes the states instantaneously and relabels them to be used as the next single support phase initial condition.

3. CONTROL ARCHITECTURE

The proposed control architecture, as illustrated in Fig. 1(d), has two modes: 1) high-level mode where the system is controlled by a dynamic based controller and 2) low-level mode where the system is controlled by a network consists of two controller nodes and a feedback node. Each controller node is attached to an active joint of the robot and solely gets position and velocity feedback from the corresponding joint in addition to the hip posture state of the robot received from the feedback node (i.e., angle of ϕ_1 stated in Fig. 1(c)).

The system is controlled by high-level controllers as long as it gets unstable using the low-level controllers. In this mode, the positions, the speeds and the controller inputs of the joints are fed into the *Learning Agents* of low-level controllers in order to train the controllers. At the same time, *Critic Agents* of low-level controllers evaluate whether they can generate the desired trajectories. 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 the *Control Agents* of the low-level controllers. A schema of this process is shown in Fig. 2.

It is desired that when the low-level controllers are active, the system acts like when it is controlled by the high-level controller. Due to decentralized nature of the low-level controllers, each node should generate the output according to the hip posture state. Therefore, the nodes are fed by the hip posture state using a feedback node (Fig. 1(d)).

By using this control architecture, the system is utilizing the robustness of the high-level controller when the system is perturbed and the simplicity and computational efficiency of low-level controllers when it is in the steady state.

3.1. High-level controller

This controller design is based on the proposed method by [14]. 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 constraints have been chosen,

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

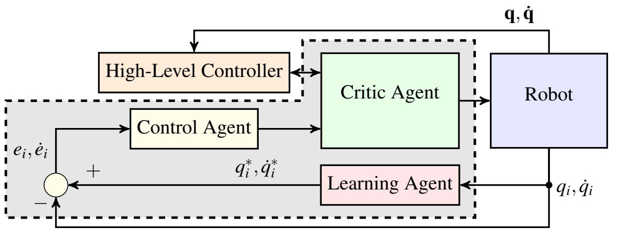


Fig. 2. Controller node structure in interaction with the robot and the high-level controller; the dashed line border area is the low-level controller boundary; critic agent decides which controller is suitable for the system.

where

$$\mathbf{h} = \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} (\theta_3 - \theta_{3d}) \\ (\phi_1 + \phi_2) \end{bmatrix}, \tag{7}$$

and $\mathbf{G}$ is a diagonal matrix define as

$$\mathbf{G} = \begin{bmatrix} g_1 & 0 \\ 0 & g_2 \end{bmatrix} \tag{8}$$

It should be noted that in (7),

$$\begin{aligned}
&x_1 = 0,\\
&z_1 = 0,\\
&x_H = l_1 \sin(\theta_1) + l_2 \sin(\theta_2),\\
&x_2 = x_H - l_1 \sin(\theta_5) - l_2 \sin(\theta_4),\\
&z_2 = z_H + l_1 \cos(\theta_5) + l_2 \cos(\theta_4),\\
&d_1 = x_H - x_1 = l_1 \sin(\theta_1) + l_2 \sin(\theta_2),\\
&d_2 = x_H - x_2 = l_1 \sin(\theta_5) + l_2 \sin(\theta_4),\\
&\phi_1 = \arctan(-d_1/(z_1 - z_H)),\\
&\phi_2 = \arctan(-d_2/(z_2 - z_H)).
\end{aligned} \tag{9}$$

As illustrated in Fig. 1(c), (x_H, z_H) and (x_2, z_2) are the hip location and the swing foot-end location with respect to the location of stance foot-end, i.e., (x_1, z_1). Moreover, in (8), the gains g_1 and g_2 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 angle (i.e., θ_{3d}), the second one makes the robot take steps and advance its hip.

To design a controller to satisfy constraints defined in (6) (drive them to zero), their governing dynamics should be derived. To do so, time derivatives of constraints are considered. Obviously, due to the second-order nature of the dynamics of the system and independence of constraints from generalized velocities, the first derivative of the constraints dynamics is not explicitly dependent on the inputs of the system. However, by using the second derivative of the constraints, their governing equation are obtained,

$$\frac{d^2\mathbf{y}}{dt^2} = L_f^2\mathbf{h}(\mathbf{q},\dot{\mathbf{q}}) + L_gL_f\mathbf{h}(\mathbf{q})\mathbf{u}, \tag{10}$$

where $L_gL_f\mathbf{h}(\mathbf{q})$ 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 [15], one can choose the following controller,

$$\mathbf{u} = (L_gL_f\mathbf{h})^{-1}(-L_f^2\mathbf{h} + \mathbf{v}). \tag{11}$$

In this case, the governing dynamics equation of closed-loop system is driven as follows:

$$\ddot{\mathbf{y}} = \mathbf{v}. \tag{12}$$

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

$$\mathbf{v} = k_1.\mathbf{y} + k_2.\dot{\mathbf{y}}, \tag{13}$$

where $k_1 < 0$ and $k_2 < 0$. This feedback structure makes the origin in the space of the constraints to be exponentially stable. Subsequently, it can be shown that under certain conditions, the trajectory of the system is a stable periodic orbit (via the Poincare map method [14]), which means that the robot is walking stably.

3.2. Low-level controller

Low level controller nodes are designed to just get position and velocity feedback from its corresponding joint. Therefore, to discuss their properties, the governing equation should be described in relative coordinate system, i.e., $(\bar{\mathbf{q}},\dot{\bar{\mathbf{q}}})$ where $\bar{\mathbf{q}} := (\mathbf{q}_u,\mathbf{q}_a)^T$. In this coordinate system, $\mathbf{q}_u = (\phi_1,\theta_2-\theta_1,\theta_4-\theta_5)^T$ represents unactuated coordinates and $\mathbf{q}_a = (\psi_1,\psi_2)^T$ describes actuated coordinates corresponding to each low-level controller. Thus, when leg number one is the stance leg, the actuated coordinates is equal to $(\theta_2+\theta_3,\theta_4+\theta_3)^T$ and it is equal to $(\theta_4+\theta_3,\theta_2+\theta_3)^T$ when leg number two is the stance leg.

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

$$\bar{\mathbf{D}}(\bar{\mathbf{q}})\ddot{\bar{\mathbf{q}}} + \bar{\mathbf{C}}(\bar{\mathbf{q}},\dot{\bar{\mathbf{q}}})\dot{\bar{\mathbf{q}}} + \bar{\mathbf{g}}(\bar{\mathbf{q}}) = \bar{\mathbf{B}}\cdot\mathbf{u}, \tag{14}$$

where it can be easily shown that $\bar{\mathbf{B}}$ has the form $\bar{\mathbf{B}} = [\mathbf{0}_{2\times3},\mathbf{I}_{2\times2}]^T$.

In the rest of this section, the components of the low-level controller are discussed. As depicted in Fig. 2, each controller node in the low-level controller network has three components as follows:

- Learning Agent whose mission is to reproduce the desired trajectory in function of the hip posture state based on the outcomes of the system when the high-level controller is active. This agent and its structure are discussed in Section 3.2.2.
- Critic Agent which is in charge of switching between states 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 the learning agents of 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 are denoted by $\mathbf{u}^*$. This conditions make the system switch to be controlled by low-level controllers.

When the system is controlled in the low-level state, the control agents are trying to eliminate the difference between the desired trajectories and the actual ones. To do so, the control agent output for the i^{th} actuated DoF is considered as follows,

$$u_i = u_i^* + \delta u_i, \tag{15}$$

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 $(\delta\bar{\mathbf{q}},\delta\dot{\bar{\mathbf{q}}})$. The governing equation for the i^{th} perturbed actuated DoF can be rewritten as

$$\bar{D}_{ii}\delta\ddot{\bar{q}}_i + \Delta_i^{(1)}\delta\dot{\bar{q}}_i + \Delta_i^{(2)}\delta\bar{q}_i + \Delta_i^{(3)} = \delta u_i, \tag{16}$$

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

Due to existence of a critic agent, the large disturbances make the system use the high-level controller instead of the low-level controller network. This means that all variables are bounded on the domain of 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 (17) for δu_i makes the system (16) stable.

$$\delta u_i = -\frac{1}{\varepsilon}\left(\delta\dot{\bar{q}}_i + c\delta\bar{q}_i\right), \tag{17}$$

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

Proof: The closed-loop equation for the i^{th} actuated DoF with the proposed controller structure (17) can be written as

$$\varepsilon\bar{D}_{ii}\delta\ddot{\bar{q}}_i + (1+\varepsilon\Delta_i^{(1)})\delta\dot{\bar{q}}_i + (c+\varepsilon\Delta_i^{(2)})\delta\bar{q}_i + \varepsilon\Delta_i^{(3)}$$

$$= 0, \tag{18}$$

Therefore, by using singular perturbation analysis [24] (see Appendix A.2), the closed-loop response will be as follows:

$$\delta\bar{q}_i(t) = \delta\bar{q}_i(0)e^{-ct} + \mathcal{O}(\varepsilon) \tag{19}$$

$$\delta\dot{\bar{q}}_i(t) = -c\,\delta\bar{q}_i(0)e^{-ct} + (\delta\dot{\bar{q}}_i(0) + c\delta\bar{q}_i(0))\delta_D(t/\varepsilon) + \mathcal{O}(\varepsilon), \tag{20}$$

where $\delta_D(\cdot)$ decays exponentially with the rate of its argument and $\delta_D(0) = 1$. □

The proposition 1 implies that the controller agent with the structure (15) and (17) guarantees the i^{th} actuated DoF achieve a trajectory within $\mathcal{O}(\varepsilon)$-neighbor of the desired trajectory.

3.2.2 Imitating/learning agent

The missions of this agent are learning and generating the desired trajectory for its corresponding DoF. This trajectory is the projection of the trajectory of system on the state space of the corresponding DoF. In order to generate it, this projection is transformed using action-angle transformation.

In this agent, the action coordinates are learned and generated based on the action coordinate via a map called *Shaping Network* described in Section 3.2.2.1. However, due to the nature of the projection, the angle coordinate should be derived based on the state space of the system.

The imposed constraints on the robot drive the system to the zero dynamics manifold. On this manifold, the dynamics of the system can be expressed by the hip posture state and the passive joints states. However, if the passive joints are sufficiently stiff, the posture state dynamics will be dominant. Hence, the system can be expressed only by the posture state and by this assumption, the posture state (i.e., ϕ_1) is monotonic in each cycle [25]. Therefore, this state will be used as the angle coordinate or the input of the Shaping Network for this agent.

Shaping Network: 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 i^{th} actuated DoF are written as follow,

$$q^*_{a,i} = \sum_{k=1}^{n} w^i_k \Phi_k(\phi_1) + w^i_0 \tag{21}$$

$$\dot{q}^*_{a,i} = \sum_{k=1}^{n} v^i_k \Phi_k(\phi_1) + v^i_0 \tag{22}$$

$$u^*_i = \sum_{k=1}^{n} \theta^i_k \Phi_k(\phi_1) + \theta^i_0, \tag{23}$$

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

$$\Phi_k(\phi_1) = \exp\left(-\frac{1}{\eta^2}(\phi_1 - \chi_k)^2\right); \tag{24}$$

ϕ_1 is the hip posture state and w^i_k, v^i_k and θ^i_k are weights of RBFs in the network.

It should be noted that in (24), χ_k is the center of radial basis function. These centers can be chosen by various methods, e.g., randomly sampled among the input instances, obtained by Orthogonal Least Square Learning Algorithm [26] or chosen via clustering of the inputs. Besides, η controls the width of RBFs which is usually considered as constant for all RBFs.

In the learning mode, the network's weights are adaptively changed in order to minimize the following objective function

$$J_i(t) = \frac{1}{2}(q^*_{a,i}(t) - q_{a,i}(t))^2 + \frac{1}{2}(\dot{q}^*_{a,i}(t) - \dot{q}_{a,i}(t))^2 + \frac{1}{2}(u^*_i(t) - u_i(t))^2. \tag{25}$$

Using gradient descent method [27] to derive the adaptive laws for the weights in the network yields,

$$\dot{w}^i_0 = -\gamma_w \nabla_{w^i_0} J_i = -\gamma_w (q^*_{a,i} - q_{a,i}), \tag{26}$$

$$\dot{w}^i_k = -\gamma_w \nabla_{w^i_k} J_i = -\gamma_w (q^*_{a,i} - q_{a,i})\Phi_k(\phi_1), \quad k \in \{1,2,...,n\}, \tag{27}$$

$$\dot{v}^i_0 = -\gamma_v \nabla_{v^i_0} J_i = -\gamma_v (\dot{q}^*_{a,i} - \dot{q}_{a,i}), \tag{28}$$

$$\dot{v}^i_k = -\gamma_v \nabla_{v^i_k} J_i = -\gamma_v (\dot{q}^*_{a,i} - \dot{q}_{a,i})\Phi_k(\phi_1), \quad k \in \{1,2,...,n\}, \tag{29}$$

$$\dot{\theta}^i_0 = -\gamma_\theta \nabla_{\theta^i_0} J_i = -\gamma_\theta (u^*_i - u_i), \tag{30}$$

$$\dot{\theta}^i_k = -\gamma_\theta \nabla_{\theta^i_k} J_i = -\gamma_\theta (u^*_i - u_i)\Phi_k(\phi_1), \quad k \in \{1,2,...,n\}, \tag{31}$$

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 they will be used to generate the desired trajectory and control input.

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 by using the error between the output of the Shaping Network and the outputs of the system. For each controller node, the error is defined as follows:

$$e_i = (q_{a,i} - q^*_{a,i}), \tag{32}$$

$$\dot{e}_i = (\dot{q}_{a,i} - \dot{q}^*_{a,i}), \tag{33}$$

which will be used to generate the *Assessment Value* defined as follows:

$$u_{\varepsilon,i} = \begin{cases} 1 & e_{\varepsilon,i} > e_{max} \\ 0 & \text{else,} \end{cases} \tag{34}$$

where

$$e_{\varepsilon,i} = k_e|e_i| + k_{\dot{e}}|\dot{e}_i|, \quad i = 1,2. \tag{35}$$

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*,

$$\dot{\varepsilon}_i = -\frac{1}{\varepsilon_l}(1 - u_{\varepsilon,i})\varepsilon_i + \varepsilon_u u_{\varepsilon,i}(1 - \varepsilon_i), \tag{36}$$

where ε_l and ε_u are small constants and ε_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 in a fast manner when the error is greater than a threshold. In other word, 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 (36) 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. Hence, when the output reaches a certain level, the imitating agent generates the desired trajectory within an acceptable error margin at all times.

At last, based on the output of the Assessment System and its current state, the critic agent will make the decision about the control ability of low-level controller. In order to avoid Zeno phenomenon and oscillation in switching between high-level controller 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 high-level controller and $0 < \varepsilon_i < E_u$, 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. After that, if $\varepsilon_i > E_l > E_u$, 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.

3.3. Architecture summary

As a summary, the structure of low-level controller node is depicted in Fig. 3. In this structure, the control input of the each actuated DoF is determined by the corresponding critic agent. When the system is not trained, the control action of the high-level controller 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 cut the high-level controller action and reroute the input of the system to the output of the control-agent.

Table 1. Mechanical Parameters of the modeled robot (see Fig. 1(a) and Fig. 1(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 $\cdot$ m^2)	0.93	1.08	2.22

4. SIMULATION

In this section, simulation of a robot prototype called RABBIT [16] with passive knee joints has been considered. The mechanical parameters of this robot is presented in Table 1. In addition to that, spring constant and damping constant of the knee joints are equal to $\kappa = 200$ and $\beta = 40$, respectively. Then, in order to construct the virtual constraints mentioned in (8) and (9), it is desired that $g_1 = 62.5$, $g_2 = 500$ and $\theta_{3d} = -\pi/9$ rad. Also, to stabilize the virtual constraints, feedback gains are set to $k_1 = 9000$ and $k_2 = 400$.

To find the approximation of the desired trajectories, the number of RFBs, n, is set to 100. Also, the width of the radial basis function, η is considered as 0.1. In addition to that, all gradient descent gains (i.e., γ_w, γ_v, γ_θ, γ_μ), are set to 10. When the system is switched to be controlled by low-level controllers, the controller agents are using the value of $1/300$ for the inverse of velocity feedback gain (i.e., ε) and 50 for the ratio of the position feedback gain to the velocity feedback gain(i.e., c). In this simulation, all critic agents are using the parameters' values stated in Table 2.

By using described parameters' values, the result of the simulation for more than 100 walking steps is depicted in the following figures. During the training, the error between 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. In Fig. 4, the position errors of the actuated DoFs are depicted versus time. When one of the robot's legs hits the ground, the Shaping Networks cannot approximate the outputs accurately. Therefore, the errors increase at these times. Note that these figures show the system with the low-level controllers is stable within a bounded error.

As mentioned, the high-level controller is working based on regulating the defined virtual constraints. In Fig. 5, the performance of the system using the high-level con-

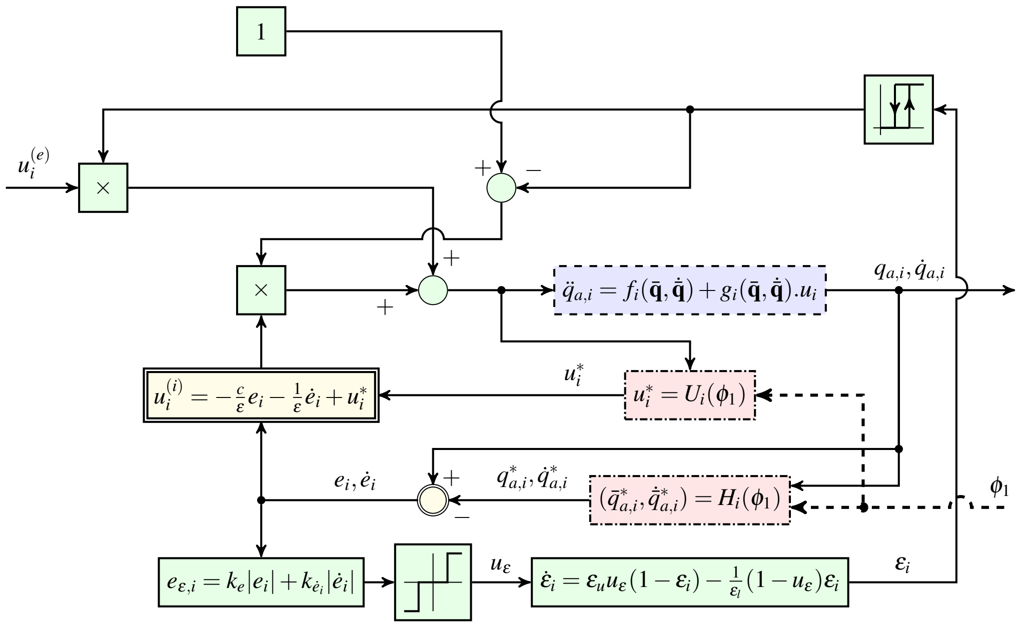

1
$u_i^{(e)}$
×
+
−
+
+
$\ddot{q}_{a,i} = f_i(\bar{\mathbf{q}},\dot{\bar{\mathbf{q}}}) + g_i(\bar{\mathbf{q}},\dot{\bar{\mathbf{q}}}).u_i$
$q_{a,i},\dot{q}_{a,i}$
u_i^*
$u_i^{(i)} = -\frac{c}{\varepsilon}e_i - \frac{1}{\varepsilon}\dot{e}_i + u_i^*$
$u_i^* = U_i(\phi_1)$
$e_i,\dot{e}_i$
$q^*_{a,i},\dot{q}^*_{a,i}$
$(\bar{q}^*_{a,i},\dot{\bar{q}}^*_{a,i}) = H_i(\phi_1)$
ϕ_1
$e_{\varepsilon,i} = k_e|e_i| + k_{\dot{e}_i}|\dot{e}_i|$
u_ε
$\dot{\varepsilon}_i = \varepsilon_u u_\varepsilon(1-\varepsilon_i) - \frac{1}{\varepsilon_l}(1-u_\varepsilon)\varepsilon_i$
ε_i

Fig. 3. Implementation of the low-level controller for the ith actuated DoF. Single border blocks (green blocks in colored version) represents subsystems of critic agent; 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 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.

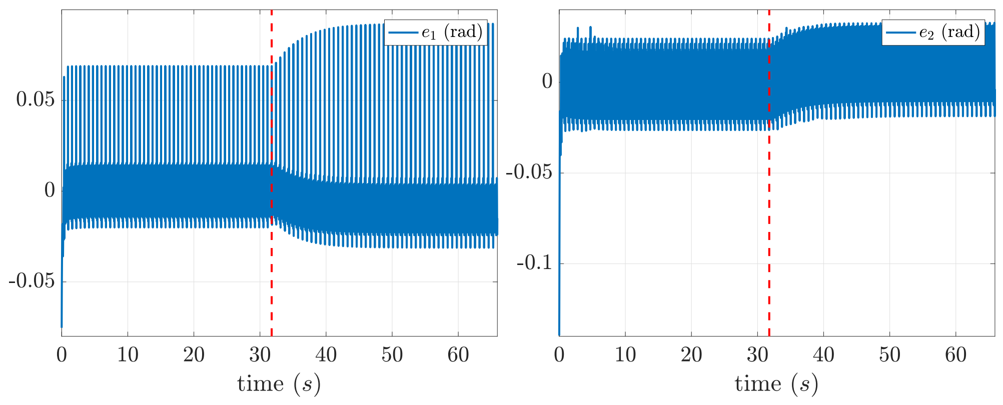


Fig. 4. Error of the imitating agents' outputs for controller nodes, i.e., $e_i = \psi_i - q^*_{a,i}$ for $i = 1,2$. The red dashed line shows the time the system switched to be controlled by low-level controllers network.

troller and the low-level controllers is depicted.

At last, the simulated gait is depicted in Fig. 6.

4.1. Performance in the presence of disturbance

The robustness of the closed-loop system in the presence of external disturbances is one of the key elements in the usability of such controllers. If the low-level controllers is not robust enough, the system gets unstable quickly after it is switched to be controlled by the low-level controllers. Hence, the critic agents put the high-level controller back in charge as soon as possible which will be in contradiction with the considered objectives.

The under study system is an under-actuated unstable dynamical system which contains impact events. Therefore, if the closed-loop system is not robust enough, the system will get unstable even in absence of any disturbances. However, the simulation conducted in Section 4 shows that the system is stable when it is under the control of low-level controllers. It can be concluded that the system will be robust in the presence of disturbances. This claim has been investigated by conducting 3 different simulations. At the first simulation, a moderate disturbance

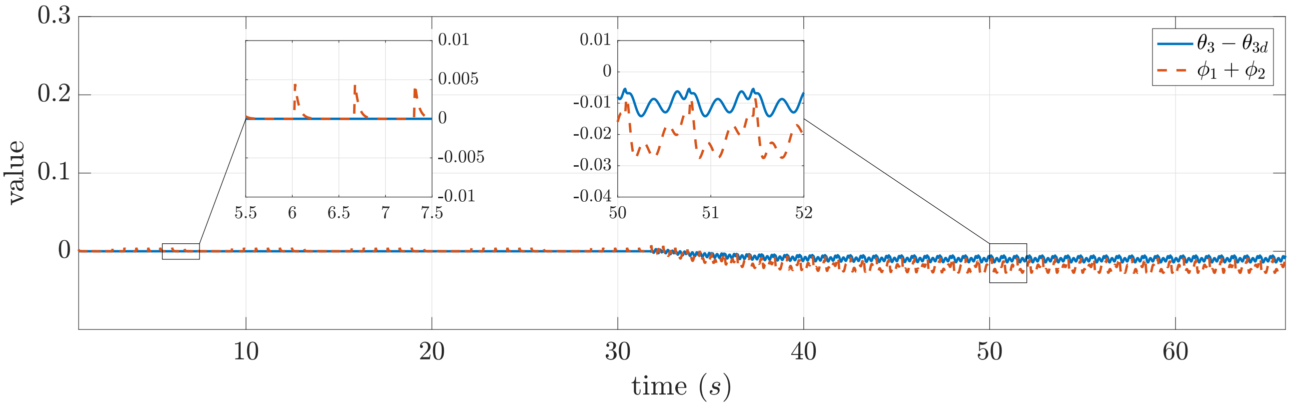


Fig. 5. Time evolution of the constraints' values defined in (6).

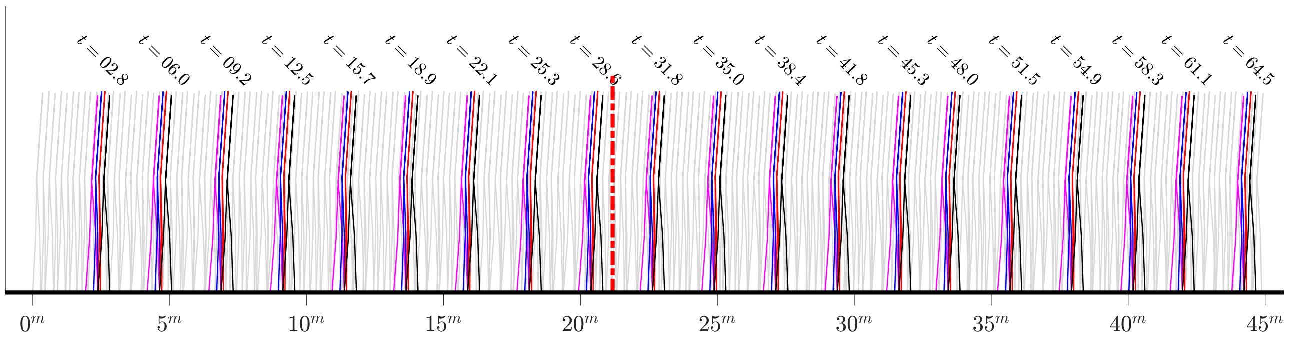
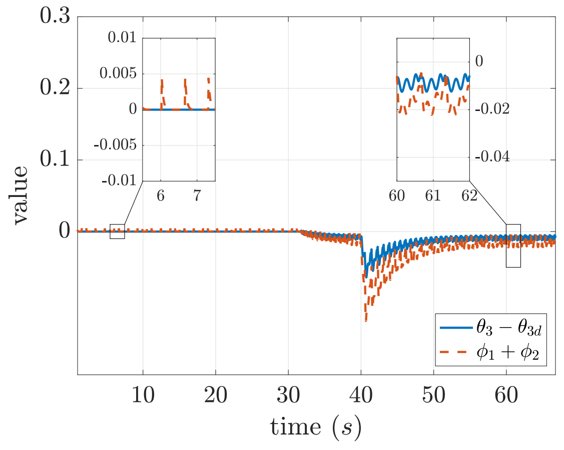
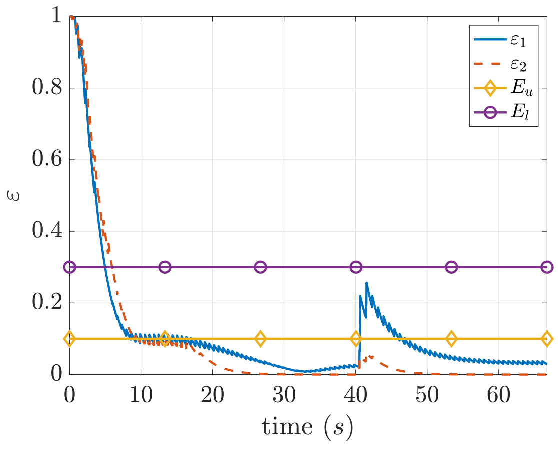


Fig. 6. Simulated gait; the divider dashed red line shows the position that the system has been switched to be controlled by the low-level controllers.



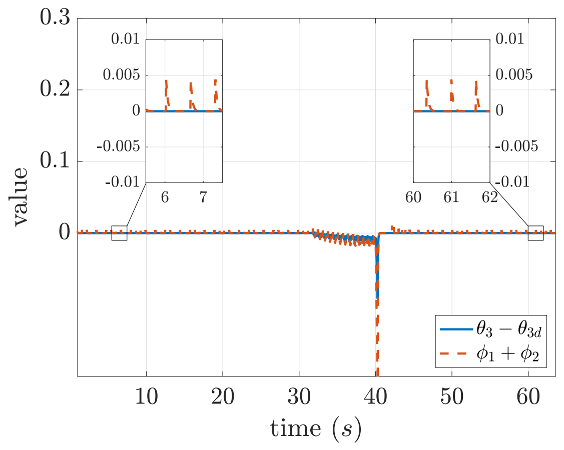
(a) Time evolution of the constraints' values.



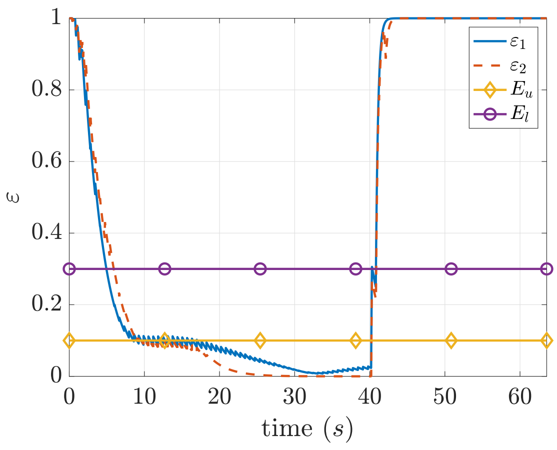
(b) Time evolution of the critic values.

Fig. 7. The system output when it is undergone a moderate disturbance impulse at time $t = 40^s$; the impulse amplitude and its duration were 3^{Nm} and 0.1^s respectively.

impulse force with the amplitude $\delta = 3^{Nm}$ and the duration $\tau = 0.1^s$ has been exerted on the system at the time $t = 40^s$. The results are depicted in Fig. 7. As it is shown, the virtual constraints have been violated after the system gets disturbed. It leads to the increase of the critic agents' values. However, the low-level controllers suppress the input disturbance afterwards. At the second conducted simulation, a large disturbance impulse force with the amplitude $\delta = 10^{Nm}$ and the duration of $\tau = 0.1^s$ has been exerted on the system at the time $t = 40^s$. The results are

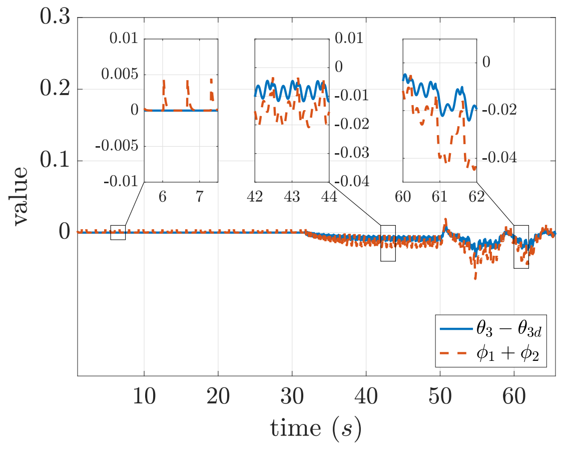


(a) Time evolution of the constraints' values.

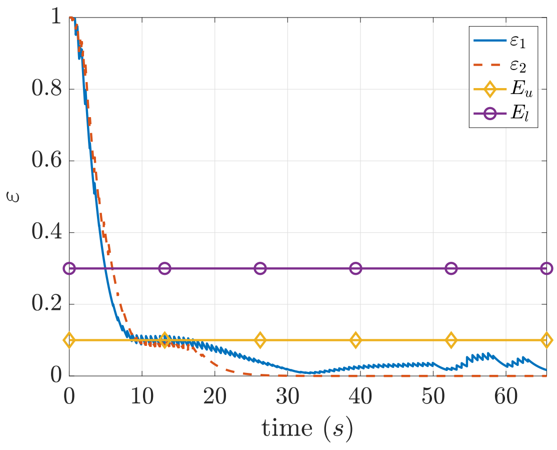


(b) Time evolution of the critic values.

Fig. 8. The system output when it is undergone a moderate disturbance impulse at time $t = 40^s$; the impulse amplitude and its duration were 10^{Nm} and 0.1^s, respectively.



(a) Time evolution of the constraints' values.



(b) Time evolution of the critic values

Fig. 9. The system output when it is undergone a uniformly distributed disturbance after time $t = 50^s$; the maximum amplitude of the disturbance is 1^{Nm}.

shown in Fig. 8. In this situation, the disturbance is so large that it makes the critic values get larger than the upper threshold, i.e. E_u. Therefore, the system is switched to be controlled by high-level controller which makes the system stable by using the global feedbacks. It should be mentioned that in this case, the low-level controllers switch back into the training state. At last, to show the performance of the system with the low-level controllers undergone continuous disturbances, a simulation is conducted with a uniformly distributed disturbance with the maximum amplitude $A_{max} = 1^{Nm}$ and it has been imposed on the system after $t = 50^s$. The results are illustrated in Fig. 9. As it is shown, the constraints values have been increased. However, the system does not get unstable (the critics values remain less than E_u).

Therefore, the proposed architecture is robust in the presence of small to moderate disturbances (impulse or continuous ones) and it makes the system switch to robustly designed high-level controller in the presence of large disturbances.

5. CONCLUSION

In this study, a model for the control of under-actuated biped robots is presented which considers the hierarchy of nervous system and its learning capability under the conscious trainings in the human. The presented framework consists of two independent layers: 1) the High-Level

Table 2. Values of the critic agents' parameters.

Param.	Description	Value
e_{max}	maximum allowable error (see (34))	0.1
k_e	error gain (see (35))	1
$k_{\dot{e}}$	error velocity gain (see (35))	0
ε_l	constant gain controls the fast dynamic time constant in (36)	0.4
ε_u	constant gain controls the slow dynamic time constant in (36)	0.3
E_u	the level of the switching from the high-layer controller to the low-level controllers	0.01
E_l	the level of the switching from the low-level controllers to the high-layer controller	0.30

Controller and 2) a network of Low-Level Controllers. The high-level controller works as the brain cortex and control the system whenever the low-level distributed controllers (inspired from the spinal cord and the motor neurons), are not trained enough or the system gets unstable under their commands (due to the large disturbances and lack of global feedback). In this state, low-level controllers get trained and when they are ready, the high-level controller is turned off and the system is controlled by the low-level controllers. Interestingly, the low-level controllers can stabilize one of the most unstable and dynamic phenomena in the world just by using the hip posture state and the local states from their corresponding joint as feedback and without any data from other joints. This framework makes the control of the system modular, simple and computationally efficient.

Although the simulation conducted in this manuscript is based on the parameters of a real robotic platform (RABBIT biped robot), one of the most important part of this research is implementing and testing this architecture on a real robotic platform. It is considered as the next step for the current study.

At last, it is worth mentioning that the presented framework is quite general and can be used to control any rhythmic activity in fully-actuated or under-actuated systems, even unstable ones in a decentralized fashion.

APPENDIX A

A.1. Definition of Parameters in the Perturbed Dynamics

In this section, the variables used in (16) are given by

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

where C_{ij} and D_{ij} is denoting the element of row i and column j of the matrix $\mathbf{C}$ and the matrix $\mathbf{D}$, respectively. Also, g_i denotes the i^{th} component of the vector $\mathbf{g}$. Moreover,

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

A.2. Perturbation analysis of dynamics of an actuated DoF with high-gain controller

In this section, singular perturbation analysis of (18) is considered.

Let us assume that there exists a regular expansion for the solution, i.e.,

$$\delta\bar{q}_i(t) = \delta Q_i^{(0)}(t/\varepsilon) + \varepsilon\delta Q_i^{(1)}(t/\varepsilon) + \cdots \tag{A.3}$$
$$+ \delta q_i^{(0)}(t) + \varepsilon\delta q_i^{(1)}(t) + \cdots. \tag{A.4}$$

It should be noted that due to the singularity of the system, $Q_i^{(n)}(t/\varepsilon)$ for $n = 0, 1, \cdots$ are added to the solution to prevent the boundary layer jump near $t = 0$. Substituting the proposed expansion into (18) yields

$$\begin{cases} \delta\dot{q}_i^{(0)} + \mathrm{c}\delta q_i^{(0)} = 0, \\ \delta\dot{q}_i^{(1)} + \mathrm{c}\delta q_i^{(1)} = -\Delta_i^{(3)} - \bar{D}_{ii}\delta\ddot{q}_i^{(0)} - \\ \Delta_i^{(1)}\delta\dot{q}_i^{(0)} - \Delta_i^{(2)}\delta q_i^{(0)} \\ \delta\dot{q}_i^{(n)} + \mathrm{c}\delta q_i^{(n)} = -\bar{D}_{ii}\delta\ddot{q}_i^{(n-1)} - \Delta_i^{(1)}\delta\dot{q}_i^{(n-1)} - \\ \Delta_i^{(2)}\delta q_i^{(n-1)}, \quad n \in \{2, 3, \cdots\}. \end{cases} \tag{A.5}$$

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

$$\begin{cases} \delta q_i^{(0)}(t) = \mathrm{A}e^{-\mathrm{c}t} \\ \delta\dot{q}_i^{(0)}(t) = -\mathrm{cA}e^{-\mathrm{c}t}, \end{cases} \tag{A.6}$$

where A is a constant determined by the initial conditions.

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

$$\begin{cases} \bar{D}_{ii}\delta\ddot{Q}_i^{(0)} + \delta\dot{Q}_i^{(0)} = 0, \\ \bar{D}_{ii}\delta\ddot{Q}_i^{(1)} + \delta\dot{Q}_i^{(1)} = -c\delta Q_i^{(0)} - \Delta_i^{(1)}\delta\dot{Q}_i^{(0)}, \\ \bar{D}_{ii}\delta\ddot{Q}_i^{(2)} + \delta\dot{Q}_i^{(2)} = -c\delta Q_i^{(1)} - \Delta_i^{(1)}\delta\dot{Q}_i^{(1)} \\ \qquad\qquad -\Delta_i^{(2)}Q_i^{(0)} - \Delta_i^{(3)}, \\ \bar{D}_{ii}\delta\ddot{Q}_i^{(n)} + \delta\dot{Q}_i^{(n)} = -c\delta Q_i^{(n-1)} - \Delta_i^{(n-1)}\delta\dot{Q}_i^{(n-1)} \\ \qquad\qquad -\Delta_i^{(2)}Q_i^{(n-2)}, \quad n \in \{3, 4, \cdots\}. \end{cases} \tag{A.7}$$

In these equations, $\bar{D}_{ii}$'s have positive upper and lower bounds ($\bar{\mathbf{D}}(t)$ is a positive definite matrix for $\forall t \in \mathbb{R}$), therefore the lowest order term of the boundary layer solution is exponentially stable [28, p. 154] and decay with the rate t/ε,

$$\begin{cases} \delta\dot{Q}_i^{(0)}(t/\varepsilon) = \mathrm{B}\delta_D(t/\varepsilon), \\ \delta Q_i^{(0)}(t/\varepsilon) = \varepsilon\mathrm{B}\int_0^{t/\varepsilon} \delta_D(\tau)\mathrm{d}\tau. \end{cases} \tag{A.8}$$

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:

$$\begin{cases} \delta\bar{q}_i(t) = \mathrm{A}e^{-ct} + \mathcal{O}(\varepsilon), \\ \delta\dot{\bar{q}}_i(t) = -c\mathrm{A}e^{-ct} + B\delta_D(t/\varepsilon) + \mathcal{O}(\varepsilon), \end{cases} \tag{A.9}$$

Hence, satisfying the initial conditions yields,

$$\begin{aligned} &\mathrm{A} = \delta\bar{q}_i(0), \\ &\mathrm{B} = \delta\dot{\bar{q}}_i(0) + c\delta q_i(0). \end{aligned} \tag{A.10}$$

REFERENCES

[1] D. Kulic, G. Venture, K. Yamane, E. Demircan, I. Mizuuchi, and K. Mombaur, "Anthropomorphic movement analysis and synthesis: a survey of methods and applications," *IEEE Transactions on Robotics*, vol. 32, pp. 776-795, Aug. 2016.

[2] Y. Hurmuzlu, F. Génot, and B. Brogliato, "Modeling, stability and control of biped robots—a general framework," *Automatica*, vol. 40, pp. 1647-1664, Oct. 2004.

[3] M. Vukobratović, B. Borovac, D. Surla, and D. Stokic, *Biped Locomotion*, Dynamics, Stability, Control and Application, Springer Science & Business Media, Berlin, Heidelberg, 1990.

[4] A. M. Khan, D.-w. Yun, M. A. Ali, K. M. Zuhaib, C. Yuan, J. Iqbal, J. Han, K. Shin, and C. Han, "Passivity based adaptive control for upper extremity assist exoskeleton," *International Journal of Control, Automation and Systems*, vol. 14, pp. 291-300, Feb 2016.

[5] J. W. Grizzle, C. Chevallereau, R. W. Sinnet, and A. D. Ames, "Models, feedback control, and open problems of 3D bipedal robotic walking," *Automatica*, vol. 50, no. 8, pp. 1955-1988, 2014.

[6] T. Luksch, *Human-like Control of Dynamically Walking Bipedal Robots*, Ph.D. Thesis, University of Kaiserslautern, Kaiserslautern, 2010.

[7] N. M. Bora, G. V. Molke, and H. R. Munot, "Understanding human gait: a survey of traits for biometrics and biomedical applications," *Proc. of International Conference on Energy Systems and Applications*, IEEE, pp. 723-728, 2015.

[8] C. Chevallereau and Y. Aoustin, "Optimal reference trajectories for walking and running of a biped robot," *Robotica*, vol. 19, pp. 557-569, Sept. 2001.

[9] R. Heydari and M. Farrokhi, "Robust model predictive control of biped robots with adaptive on-line gait generation," *International Journal of Control, Automation and Systems*, vol. 15, pp. 329-344, Feb 2017.

[10] J. Cronin, R. Frost, and R. Willgoss, "Walking biped robot with distributed hierarchical control system," *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA '99)*, IEEE, pp. 150-156, 1999.

[11] T. Odashima, Z. Luo, and S. Hosoe, "Hierarchical control structure of a multilegged robot for environmental adaptive locomotion," *Artificial Life and Robotics*, vol. 6, pp. 44-51, March 2002.

[12] P. Arena, L. Fortuna, M. Frasca, and G. Sicurella, "An adaptive, self-organizing dynamical system for hierarchical control of bio-inspired locomotion," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, pp. 1823-1837, August 2004.

[13] J. H. Barron-Zambrano, C. Torres-Huitzil, and B. Girau, "Perception-driven adaptive CPG-based locomotion for hexapod robots," *Neurocomputing*, vol. 170, pp. 63-78, 12 2015.

[14] J. W. Grizzle, G. Abba, and F. Plestan, "Asymptotically stable walking for biped robots: analysis via systems with impulse effects," *IEEE Transactions on Automatic Control*, vol. 46, no. 1, pp. 51-64, 2001.

[15] F. Plestan, J. W. Grizzle, E. R. Westervelt, and G. Abba, "Stable walking of a 7-DOF biped robot," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 4, pp. 653-668, 2003.

[16] E. R. Westervelt, J. W. Grizzle, and D. E. Koditschek, "Hybrid zero dynamics of planar biped walkers," *IEEE Transactions on Automatic Control*, vol. 48, pp. 42-56, Jan. 2003.

[17] Y. Hurmuzlu, "Dynamics of bipedal gait part II-stability analysis of a planar five-link biped," *Journal of Applied Mechanics*, vol. 60, pp. 337-343, June 1993.

[18] D. Djoudi, C. Chevallereau, and Y. Aoustin, "Optimal reference motions for walking of a biped robot," *Proceedings of IEEE International Conference on Robotics and Automation (ICRA 2005)*, IEEE, pp. 2002-2007, 2005.

[19] M. Hardt, K. Kreutz-Delgado, and J. W. Helton, "Optimal biped walking with a complete dynamical model," *Proceedings of the 38th IEEE Conference on Decision and Control*, IEEE, pp. 2999-3004, 1999.

[20] A. C. de Pina Filho, M. S. Dutra, and L. Santos, "Modelling of bipedal robots using coupled nonlinear oscillators," *Mobile Robots towards New Applications* (A. Lazinica, ed.), ch. 4, pp. 55-78, InTech, 2006.

[21] T. Buschmann, A. Ewald, A. von Twickel, and A. Büschges, "Controlling legs for locomotion-insights from robotics and neurobiology.," *Bioinspiration and Biomimetics*, vol. 10, p. 041001, June 2015.

[22] E. R. Westervelt and J. W. Grizzle, "Design of asymptotically stable walking for a 5-link planar biped walker via optimization," *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '02)*, IEEE, pp. 3117-3122, 2002.

[23] C. Liu and J. Su, "Biped walking control using offline and online optimization," *Proc. of 30th Chinese Control Conference (CCC)*, Yantai), pp. 3472-3477, IEEE, 2011.

[24] F. Verhulst, *Methods and Applications of Singular Perturbations*, vol. 50 of *Boundary Layers and Multiple Timescale Dynamics*, Springer Science & Business Media, New York, NY, June 2005.

[25] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi, and B. Morris, *Feedback Control of Dynamic Bipedal Robot Locomotion*, CRC Press, June 2007.

[26] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 2, pp. 302-309, Mar. 1991.

[27] D. Saad, *On-Line Learning in Neural Networks*, Cambridge University Press, July 2009.

[28] H. K. Khalil, *Nonlinear Systems*, Pearson Education, Prentice Hall, 2002.

Masoud Yazdani received his B.S. and M.Sc. degrees in mechanical engineering from Sharif University, Tehran, Iran, in 2008 and 2010 respectively. He is currently pursuing a Ph.D. degree in 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 in mechanical engineering and also pure mathematics from Sharif University of Technology, Tehran, Iran in 2002. He graduated from the same university with M.Sc and Ph.D. degrees in mechanical engineering in 2004 and 2008. At present, he is a professor in mechanical engineering at Sharif University of Technology. His fields of research are dynamical systems, control theory and stochastic systems.

Mahmood Saadat Foumani received his Ph.D. degree in Mechanical Engineering from Sharif University of Technology, Tehran, Iran in 2002. He was a Faculty member at Semnan University from 2002 to 2006 and is now a faculty member of 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 and chassis design and advanced mathematics.