AUT Journal of Modeling and Simulation

AUT J. Model. Simul., 56(2) (2024) 185-198
DOI: 10.22060/miscj.2025.23344.5368

A Simplified Event-based Impulsive Control Approach for Stable, Efficient, and Robust Locomotion Using Deep Reinforcement Learning

Bahareh Sadat Mortazavi, Rezvan Nasiri*, Majid Nili Ahmadabadi

Department of Electrical and Computer Engineering, Faculty of Engineering, University of Tehran, Tehran, Iran.

ABSTRACT: Biological evidence indicates that the actuation system in humans and legged animals is characterized by impulsiveness rather than continuity; i.e., control actions are concentrated within a specific phase of the motion cycle (the stance phase), while the rest of the cycle is passive. Based on this observation, we propose a simple event-based impulsive controller to generate walking cycles for legged robots. To improve optimization speed, we parametrize the controller-applied forces as a Gaussian function of time and employ a deep reinforcement learning method to optimize the controller parameters. To further enhance learning speed, an autoencoder is utilized to address the high dimensionality in the state space. Additionally, we employ a three-phase reward-shaping approach to further improve learning speed and achieve better results. In phase one, the reward function focuses on stability and forward motion to learn stable locomotion. In phase two, the reward function is modified to achieve stable locomotion with lower control effort and desired forward velocity. In phase three, the reward function remains the same as in phase two but places more emphasis on forward velocity regulation. The proposed controller, state encoder, and learning process can be implemented on a group of legged robots with actuation at the leg contact point with the ground. In this paper, the proposed approach is tested on a simulated single-legged robot. In addition, the controller robustness is analyzed considering different types of external disturbances. The simulation results indicate the efficacy of the proposed method as a bio-inspired control approach for legged locomotion.

Review History:

Received: Jul. 14, 2025
Revised: Nov. 25, 2024
Accepted: Nov. 26, 2024
Available Online: Dec. 24, 2024

Keywords:

Deep Reinforcement Learning
Event-based Control
Impulsive Control
Legged Robot

1- Introduction

For the past several years, researchers have been studying diverse control strategies for legged robots, aiming to unlock their full potential in terms of adaptability, stability, robustness, efficiency, and forward velocity. Given the intrinsic nonlinear, hybrid, discontinuous, and uncertain nature of legged locomotion, formulating an effective control strategy that embodies simplicity, efficiency, stability, and robustness is a significant challenge [1]. Consequently, devising a comprehensive controller design methodology for legged robots remains intricate and demanding. To achieve this goal, researchers have explored various avenues, including analytical approaches [2], energy-efficient controller design [3], impedance control [4], natural dynamic exploitation [5], and trajectory adaptation [6].

Nature has served as a wellspring of inspiration for numerous research endeavors in this field. Accordingly, many attempts have been made to design controllers based on biological evidence; one notable method is using central pattern generators (CPGs) as controllers [7-9]. For instance, [10] proposes a controller based on the encoded activation patterns observed in the spinal cords of salamanders. Another avenue of exploration involves impedance controllers [4]. These controllers are designed based on findings suggesting that humans modulate the impedance of their ankle joints to attain stability and efficiency during walking tasks [11-13]. Furthermore, an impressive instance is the work by [14], wherein an event-based muscle-level controller is presented.

It has been investigated that the human actuation system is impulsive [15] and event-based [16]. [15] showed that large cursorial animals, such as horses, rely on a catapult mechanism for rapid acceleration and preparing for the next stance phase. In addition, [17], [18] showed that the H-reflex activates muscles with a 200ms time delay in response to external disturbances. Moreover, other research has shown that after the push-off moment, the rest of the walking cycle (between toe-off and heel-strike) is passive [19], [20]; i.e., lower limb dynamics during the swing are governed by natural dynamics. In conclusion, the control strategy for legged animals is impulsive, event-based, and built upon the natural dynamics of the system.

Inspired by these biological facts, in [21], we presented a concurrent analytical design of a controller and passive elements (i.e., spring and damper) for impulsive actuation

*Corresponding author’s email: rezvan.nasiri@ut.ac.ir systems to generate rhythmic walking patterns. However, the proposed controller suffered from a lack of state feedback, non-smooth actuation patterns, and non-optimal timing of actuation impulses. In this paper, we address these issues by optimizing the timing of actuation, smoothing the actuation impulses, reducing the number of actuations to one per motion period, and incorporating state feedback to improve the controller's robustness to external disturbances. This approach results in a simple, impulsive, and event-based controller. Due to the complex and hybrid nature of walking, such controllers cannot be designed analytically; hence, a deep reinforcement learning (DRL) approach is utilized to resolve this complex and nonlinear mapping.

1- 1- Related work

Due to the complexity, non-linearity, and hybrid nature of legged locomotion, control strategies for legged robots should be nonlinear and phase-dependent; i.e., they should be event-based or discontinuous. Accordingly, many attempts have been made to enhance legged robots' performance through event-based controllers. For instance, event-based controllers for bipedal robots [22], predictive and robust controllers [23], [24], and controllers based on finite state machines (FSM) [25] have been explored. Besides event-based controllers, impulsive control strategies have also shown potential in generating bio-inspired and energy-efficient legged locomotion [21], [26], [27]. However, such controllers are mostly designed based on the dynamical model of the system and are not adaptive.

Some approaches benefit from both concepts (impulsive and event-based commands) to present a nonlinear closed-loop controller [28]. [28] proposes two types of event-triggering algorithms to generate impulsive control commands. The first algorithm is based on continuous event detection, while the second updates the impulsive inputs according to cyclic events. However, both algorithms are developed for continuous dynamical systems, which are not suitable for hybrid and discontinuous dynamics such as legged locomotion. In addition, this method requires the dynamical equations of the system which is mostly unknown in legged locomotion tasks.

Due to the ability of deep neural networks (DNN) to encode nonlinear complex relations between robot states and controller commands and its adaptation capability, this toolbox has recently been utilized in many studies involving reinforcement learning on legged locomotion [29-31]. For instance, [29] utilizes deep reinforcement learning (DRL) to generate natural walking from scratch. The problem of close-to-natural human walking is divided into three stages of learning: standing, stepping, and then gradually improving the gait. This strategy accelerates the learning process and helps the bipedal model generate close-to-natural human walking patterns. Another example is [30], which presents a two-level continuous control strategy using DRL for a 3D bipedal robot; while the low-level controller moves the joints over the desired trajectories, the high-level controller generates the optimal trajectories for the low-level controller. A similar two-level continuous control strategy is also presented in [31], where the problem of pursuing a specific goal in the environment is divided into two levels of training: (1) learning basic movements such as walking and (2) learning how to combine basic movements to achieve the final goal. However, these proposed controllers are not event-based.

As mentioned earlier, in legged locomotion, a proper control strategy should be energy-efficient, robust, and adaptive to effectively enhance the locomotion task. Hence, the controller should consume less energy, provide a high level of uncertainty robustness, and grant adaptability. Accordingly, many attempts have been made to enhance legged robots' performance by improving the control strategy [40].

From our design perspective, controllers for legged robots can be divided into four main categories: discontinuous (event-based or impulsive) and continuous controllers, each with rule-based (classic) and RL-based (data-driven) design approaches. Each category has its own advantages and disadvantages [41]. For instance, rule-based designs, which often rely on detailed environment models and robot dynamics, can deliver reliable tracking performance [42]. However, they are not robust in the face of uncertainties in robot.

dynamics and environment models [43]. Additionally, continuous controllers are generally associated with high energy consumption [44].

On the other hand, discontinuous controllers use energy in short intervals, reducing overall energy consumption at the cost of increased tracking error [21]. RL-based designs offer a model-free alternative, allowing adaptable performance without prior knowledge of the environment model or robot dynamics, reducing reliance on accurate system identification, and increasing controller robustness. However, they often require considerable learning time, especially to learn continuous actuation patterns. Combining an RL-based controller design with a discontinuous actuation pattern can minimize the search space and improve training time.

Table. 1 summarizes the advantages and disadvantages of each control category. Based on this table, the only drawback of discontinuous controllers designed with an RL-based approach is low tracking performance. Nevertheless, it is worth mentioning that in legged locomotion, tracking performance is not the main objective, neither in biology [45] nor in robotics [46].

To achieve energy efficiency and high control robustness, we propose a controller and optimization strategy using the DRL toolbox, which includes: (1) a simple event-based impulsive controller (SEBIC) architecture, (2) a state encoder using an autoencoder, and (3) a three-phase learning process that improves learning speed and results. The proposed strategy is implemented on a simulated single-legged robot, with actuators

at the contact point with the ground to modulate the ground reaction force (GRF) and attain similar GRF patterns as in legged animals [32-35].

The rest of this paper is organized as follows: The

Table 1. The overall comparison between four different control categories.

Controller design approach	**Rule-based (classic)**		**RL based (data-driven)**	
Controller type	Continuous	Discontinuous	Continuous	Discontinuous
Require system model	Yes	Yes	No	No
Energy consumption	High	Low	High	Low
Learning/training time	----	----	High	Low
Tracking error	Low	High	Low	High
Robustness	Low	Low	High	High
Adaptability	Yes	No	Yes	Yes

next section presents the problem statement and detailed formalization, including model description, reward definition, and learning strategy. Section III presents the simulation results, evaluating the quality of the trained policy for the single-legged robot in terms of optimality and stability. Finally, discussions, conclusions, and future work are provided in the last section.

2- Methods and materials

Our suggested control architecture for a legged robot is presented in Fig. 1. The main components of our controller are:

- **State Encoder:** Reduces the state space dimension and is trained individually using a batch dataset of the robot.
- **Stance Phase Detector:** Detects impact events and triggers the controller to generate impulsive control actions for push-off.
- **Passive Dynamics:** Comprising passive compliance and damper for each joint, these behave as nonlinear state feedback and are designed according to the method presented in [21].
- **Simple Event-Based Impulsive Controller (SEBIC):** Maps the robot states at the impact moment to the impulsive actions during the stance phase.

Based on our formulation, the impulsive control action ($F_{ag} = \left[F_x, F_y\right]^T \in \mathbb{R}^2$) is a two-dimensional vector in the sagittal plane applied to the ground at the contact point; i.e., the actuation system is prismatic. The control action is formulated as two Gaussian force profiles (i.e., $F_x = A_x\, exp\left(-\sigma_x \left(t-\mu_x\right)^2\right)$ and $F_y = A_y\, exp\left(-\sigma_y \left(t-\mu_y\right)^2\right)$), with their parameters ($\theta = \left[A_x, A_y, \sigma_x, \sigma_y, \mu_x, \mu_y\right]^T$ ò$\mathbb{R}^6$) determined by the DRL algorithm according to the robot states (Q_t) at the impact moments ($t = t_0^+$). The robot states at the impact moment form a vector of the robot's positions ($q = \left[\theta_{hip}, \theta_{knee}\right]^T \in \mathbb{R}^2$) and velocities ($v = \left[\omega_{hip}, \omega_{knee}, v_x, v_y\right]^T \in \mathbb{R}^4$), as shown in Fig. 2. Therefore, the problem is to learn a controller (i.e., a policy; π_θ) that maps the observed states ($Q_t = [q, v]^T \in \mathbb{R}^6$) of the leg at impact moments to the actuation impulses (F_{ag}). Due to the high dimensionality of the state space, we employed a state encoder to reduce the state space dimension, hence the reduced state space (S_t) is actually mapped to the action space (A_t) as $F_a(t) = A_t = \pi_\theta(S_t)$. The robot model, state encoder, reward function, and training process are explained in the following subsections.

2- 1- State encoder

We use a nonlinear encoder to reduce state dimensions and accelerate the learning process. Accordingly, a 4-layer autoencoder is utilized to reduce the observation dimension from 6 $\left(O_t \in \mathbb{R}^6\right)$ to 3 encoded dimensions ($S_t \in \mathbb{R}^3$) and then reconstruct the input in the output using the encoded states; see Fig. 2. The activation functions of the network layers are ReLU, Linear, ReLU, and Sigmoid, respectively. To train the autoencoder network, we generate a set of feasible impact states and use 70% for training and the remaining 30% for evaluation, achieving 97% evaluation accuracy.

2- 2- Deep reinforcement learning model

Consider Fig. 3, which illustrates our suggested SEBIC logic and training strategy. The SEBIC controller captures the robot states (positions and velocities) at the impact moments and predicts the stabilizer force profiles to stabilize the system in the next step (active mode). Besides the active mode, the rest of the cycle (passive mode) is governed by passive elements. During the active mode, the controller determines the shape and timing of the force impulses based on the observed states at the impact event (O_t). The reward function (R_t) is computed at the end of each step, and consequently, the policy $\left(\pi_\theta\right)$ and actions (A_t) are updated once per cycle. To train our SEBIC controller, an actor-critic

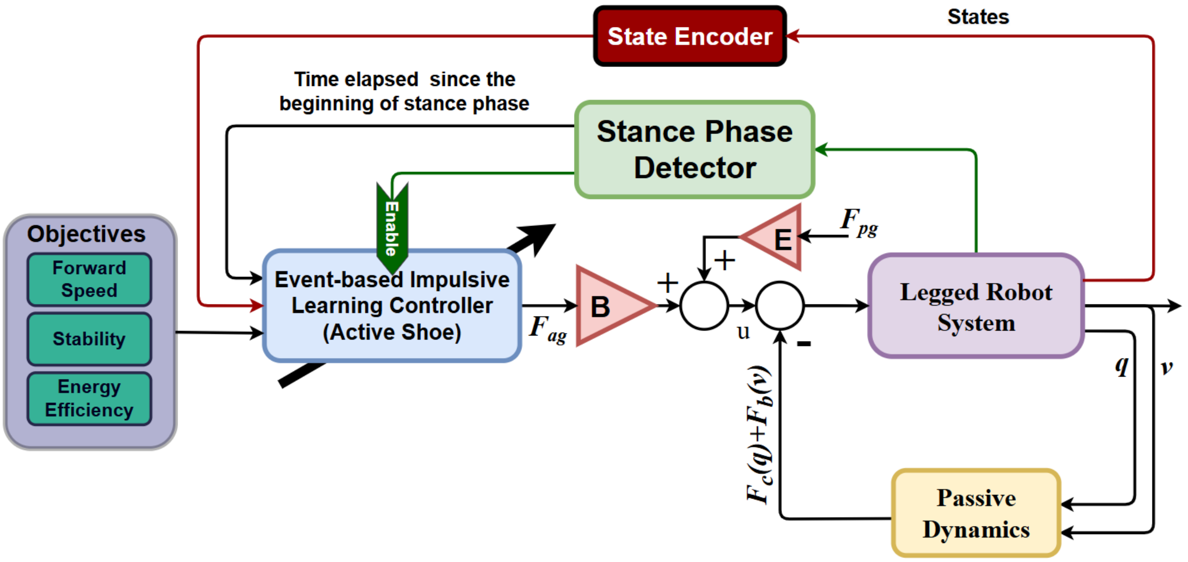


Fig. 1. The suggested control architecture for a single-legged robot employs a simple event-based impulsive controller (SEBIC). In this block diagram, a stance detector triggers the main controller (SEBIC) to apply control commands during the stance phase based on the robot's state at the impact moment, while passive dynamic feedback governs the system's natural dynamics; i.e., position (q) and velocity (v). SEBIC determines the actuation at the impact point (F_{ag}) to modulate the total ground reaction force (F_{grf}). The total GRF (F_{grf}) is the sum of the force applied by the simulated ground model (F_{pg}) and the actuators' force applied by SEBIC (F_{ag}). Moreover, E and B are the Jacobian matrices that map the F_{ag} and F_{pg} to the joint space, where $u=BF_{ag}+EF_{pg}$. In general, E and B can be different, in our simulations, they are equivalent. Also, F_c (q) and F_b (v) are passive nonlinear spring and damper forces designed based on [21], which form a nonlinear state feedback stabilizing the internal stability.

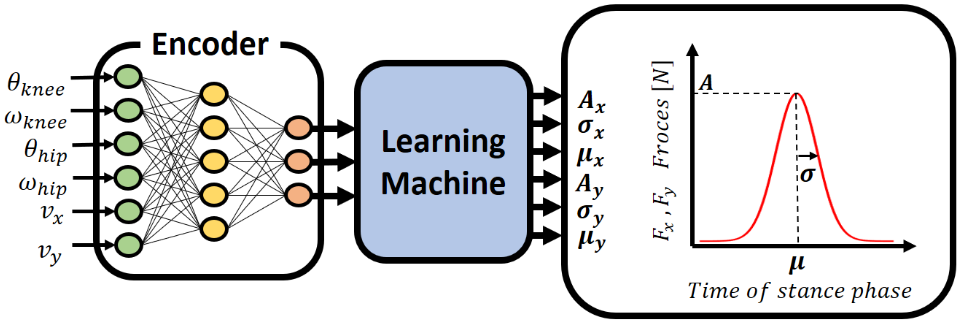


Fig. 2. The SEBIC architecture including the state encoder and machine learning model. The state encoder maps the robot's state at the impact moment (O_t ϵ R^6) to a lower dimension (S_t ϵ R^3) to facilitate the learning process. The machine learning model maps the reduced state space (S_t) to the parameters of control actions. The parameters of control action are widths (σ_x and σ_y), amplitudes (A_x and A_y), and center of Gaussian profiles (μ_x and μ_y) of forces in vertical (y) and horizontal (x) directions.

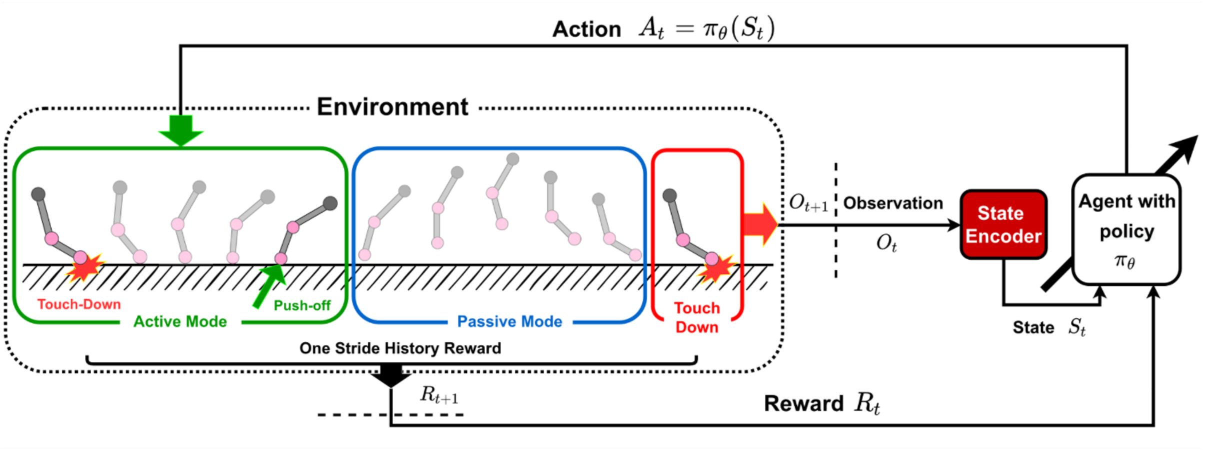


Fig. 3. Agent-environment interaction in DRL. Each locomotion stride is divided into two main modes: active (SEBIC is on) and passive (SEBIC is off). In each touch-down moment (i.e., stance phase), the active mode triggers and the encoded state data (S_t) along with a calculated reward (R_t) between previous (t) and current (t+1) touch-down moments are sent to the learning machine policy. The policy updates and sends the actions (A_t) to the environment. This action specifies the push-off force parameters (θ= [A_x,A_y,σ_x,σ_y,μ_x,μ_y]^T) during the stance phase of a stride. As the robot disconnects from the ground (i.e., flight phase), push-off force vanishes, and the robot enters the passive mode or swing phase in which receives no activation; i.e., F_x=F_y=0.

policy is developed in Python and connected to MATLAB via TCP/IP protocol. During learning, an off-policy algorithm called Twin Delayed Deep Deterministic Policy Gradient (TD3) [36] is used to update the actor-critic network.

The designed network consists of a 3-layer, 256-neuron policy network and two 3-layer, 256-neuron value networks with ReLU and tanh activation functions. The optimal total number of deep network parameters, determined through grid search, is approximately 136,500; any deviation from this value, either increasing or decreasing the parameter count, led to a reduction in the evaluation reward, ensuring that the model does not overfit. The network design is based on our experimental findings to ensure sufficient network representation capability and learning efficiency. To initiate the state and action in each training episode, we use the agent's previous experiences stored in the replay buffer for off-policy training. Additionally, since each decision-making unit in our problem is a stride (not a specific time step), we define each stable episode length as a specific number of strides.

2- 3- Reward function

To achieve stable, efficient, and robust locomotion performance, the reward function (R_t) is designed as a weighted summation of stability reward (R_s), forward motion reward (R_f), control effort-reward (R_e), and forward velocity reward (R_v); i.e., $R_t = w_s R_s + w_f R_f + w_e R_e + w_v R_v$, where $w_i > 0$ is the reward weight. The stability reward is defined as $R_s = 2/\left(\int_T f_v dt + 1\right)$, where f_v is the vertical force applied by a virtual surface ($f_v > 0$) to the hip joint to prevent it from falling below a certain height. When this reward is maximum ($R_s^{max} = 2$), the system is stable. The forward motion reward is equal to the sign of stride length; i.e., if stride length is positive (negative), the reward is plus (minus) one ($R_f = \pm 1$). The control effort penalty function is: $R_e = 2/\left(E_t + 1\right)$, where E_t is the integral of the absolute value of impulsive actuation divided by the traveled distance at each step. E_t is similar to the cost of transportation (COT) and the maximum of control effort-reward is $R_e^{max} = 2$. The forward velocity reward is defined as follows.

$$R_v = \begin{cases} -\left(v_x - v_{ref}\right)^2, & x_{toe} \leq 1 \\ \exp\left(-\left(v_x - v_{ref}\right)^2\right), & else \end{cases} \quad (1)$$

This function heavily penalizes forward velocity at the initial steps toward the desired velocity (v_{ref}) for traveled distances less than 1 unit. Once the traveled distance exceeds 1 unit, it considers forward velocity as a reward. This reward is the maximum ($R_v^{max} = 1$) for velocities equal to the desired one. The forward velocity reward function is inspired by the technique described in [29] to improve training performance.

2- 4- Training strategy

To improve training speed and performance, we employ a reward shaping approach [29] similar to the Method

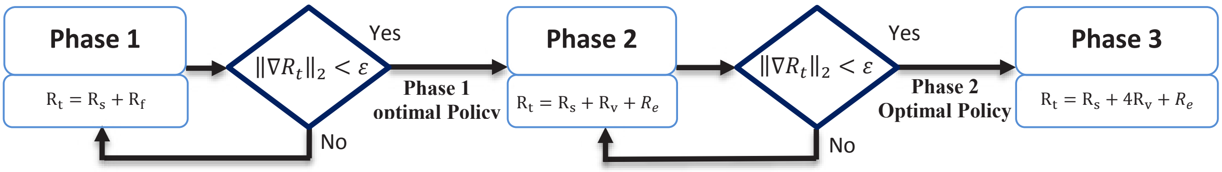


Fig. 4. Learning algorithm flowchart of the proposed training method. The training is summarized in three main steps: Phase 1: learn to move stably (R_t=R_s+R_f), Phase 2: learn to achieve higher forward velocity along with energy efficiency (R_t=R_s+R_v+R_e), and Phase 3: improve forward velocity while maintaining energy efficiency (R_t=R_s+4R_v+R_e). Each phase is initialized with the optimally trained policy from the previous phase and concludes based on the gradient criterion. The first phase is starts with a randomly generated policy.

suggested in [37]. As discussed in [37], the problem of reward shaping is an MDP if the changes in the reward function are sufficiently slower than the training speed. To achieve this, we decompose our locomotion learning problem into three phases, where the reward function changes only in these phases, and the trained policy at each phase is used as the initial policy for the next one. The reward function at each phase is reshaped based on the training goals of that phase (see Fig. 4). The first phase aims to generate a stable-walking-capable agent that walks forward stably. Accordingly, stability and forward locomotion are prioritized, while energy efficiency and forward velocity are not considered during this phase; $w_s = 1, w_f = 1, w_v = 0, w_e = 0$. The second phase focuses on speed tuning and minimizing control effort while maintaining stability. Since the forward locomotion reward is redundant with the forward velocity reward in this phase, it is omitted; $w_s = 1, w_f = 0, w_v = 1, w_e = 1$. Similarly, the third phase places more emphasis on forward velocity regulation compared to the second phase. Hence, in the third phase, the gains are the same as in the second phase but with a higher emphasis on the forward velocity reward gain; $w_s = 1, w_f = 0, w_v = 4, w_e = 1$. The learning process is formulated in Algorithm. 1, and the reward weights for each phase are summarized in Table. 2.

2- 5- Model description

The designed control architecture is implemented for a planar single-leg robot model with 2 revolute joints (see Fig. 2). The model consists of a point mass representing the coupling dynamics of the leg with the whole body, hip and knee joints, and a point foot; i.e., ($\left[\theta_{hip}, \theta_{knee}, x, y\right]^T$), indicating that the model has 4 independent coordinates and consequently 4 degrees of freedom. Joints have parallel springs and dampers designed according to [21] to govern system dynamics until the next actuation interval. To model foot-ground contact, we have utilized the soft impact model presented in [14]. Accordingly, for each leg, the state observation (O_t) has 8 dimensions, including all joint positions and velocities, and point-foot velocity and position

```
Algorithm. 1: Training
Initialize: Initial observation O0; stable episode
  stride num d; initial policy network weights π;
  for phase $i$ ∈ {1, 2, 3} do
    Initialize: Value networks weights V;
    Get $S_0$ encoding $O_0$ using state encoder;
    set $\omega_i$ according to Section 2-5;
    while True do:
      $A_t = \pi(S_t)$;
      Wait until the next touchdown;
      Get $O_{t+1}, R_{t+1}(\omega_i)$;
      Get $O_{t+1}$ encoding $S_{t+1}$;
      Get $\pi_{new}, V_{new}$ by optimizing π, V using TD3;
      t=t+1; $S_t = S_{t+1}$;
      if d ≤ t or instability occurrence then
        Select $S_0$ from prev. experiences;
        Get $O_0$ encoding $S_0$ using state decoder;
        t=0;
      end
      if converge then
        break;
      end
    end

end
```

Table 2. The reward weight values in different phases

	ω_s	ω_f	ω_v	ω_e
Phase 1	1	1	0	0
Phase 2	1	0	1	1
Phase 3	1	0	4	1

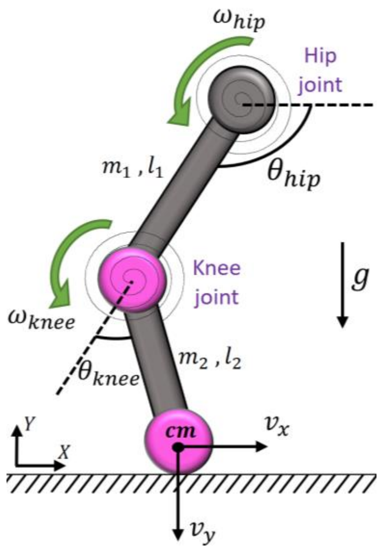


Fig. 5. The dynamical model of a representative leg of a single-legged robot is detailed in Appendix A. The measured states of the representative leg at the impact moment are $O_t=[\theta_{hip},\theta_{knee},\omega_{hip},\omega_{knee},v_x,v_y]^T$. The simulated model parameters are m_1=5kg, m_2=2kg, l_1=52cm, l_2=40cm, and g=9.81m/s^2. The passive dynamical parameters (compliance and damper coefficients) of the hip and knee joints are K_{hip}=100Nm/rad, K_{knee}=50N.m/rad, b_{hip}=2N ms/rad, and b_{knee}=5 Nms/rad.

in the x and y directions at the impact moment. However, we ignore the point-foot positions (x, y) and remove them from the state observations for the following reasons:

Since the state features are observed at every touch-down moment, the point-foot y position is always level with the ground ($y = 0$). Thus, including the y position is pointless.

As the robot moves forward on a fixed-condition ground, the x position of the pointed foot continually increases. Therefore, the x position is not a suitable parameter for discriminating between different observations.

Therefore, the state has 6 dimensions; i.e., $O_t = \left[\theta_{hip}, \theta_{knee}, \omega_{hip}, \omega_{knee}, v_x, v_y\right]^T \in \mathbb{R}^6$. We assume that all of these states are measurable with commercial sensors. For instance, $\theta_{hip}, \theta_{knee}, \omega_{hip}, \omega_{knee}$ can be measured using an encoder and v_x, v_y can be measured using IMU sensors.

3- Simulation results

In this section, we investigate the performance of the proposed learning method on a single-legged planar robot in simulation (see Fig. 2). We conducted two sets of simulations. The first set involved training the model, with the results presented in Fig. 6. The second set demonstrates the performance of the best policy, shown in Fig. 7. Finally, we applied external disturbances to the optimal policy. Since these disturbances were not part of the learning process, we effectively evaluate the robustness of the policy (see Table. 3).

3- 1- Training results

The agent rewards (i.e., total, forward velocity, and control effort) during the training process are presented in Fig. 6. Each subplot consists of three main phases, as explained in Section II-B, with different reward weights.

Fig. 6a illustrates the overall reward. It is clear that the overall reward settles before the start of new phases, indicating that learning in each phase converges to a local optimum. Fig. 6b shows the contribution of the forward velocity reward (R_v) defined in Eq. 1 with $v_{ref} = 1.5 m/s$ to the overall reward (R_t). The first phase does not include any reward term related to velocity ($w_v = 0$); consequently, the contribution of the forward velocity reward in phase one is zero. However, the second and third phases include the forward velocity reward with different coefficients ($w_v = 1$) in phase two and ($w_v = 4$) in phase three, as explained in Section II-B, resulting in a significantly higher forward velocity reward in the third phase compared to the second phase. Based on the results in this figure and the reward function weights, it can be inferred that increasing the speed reward coefficient in the third phase tunes the velocity more precisely. Additionally, the learning speed has increased significantly compared to the first two phases.

Fig. 6c represents the contribution of the control effort-reward (R_e) to the overall reward (R_t). In the first phase, due to the zero control effort-reward weight ($w_e = 0$), the contribution of control effort is zero. In the second and

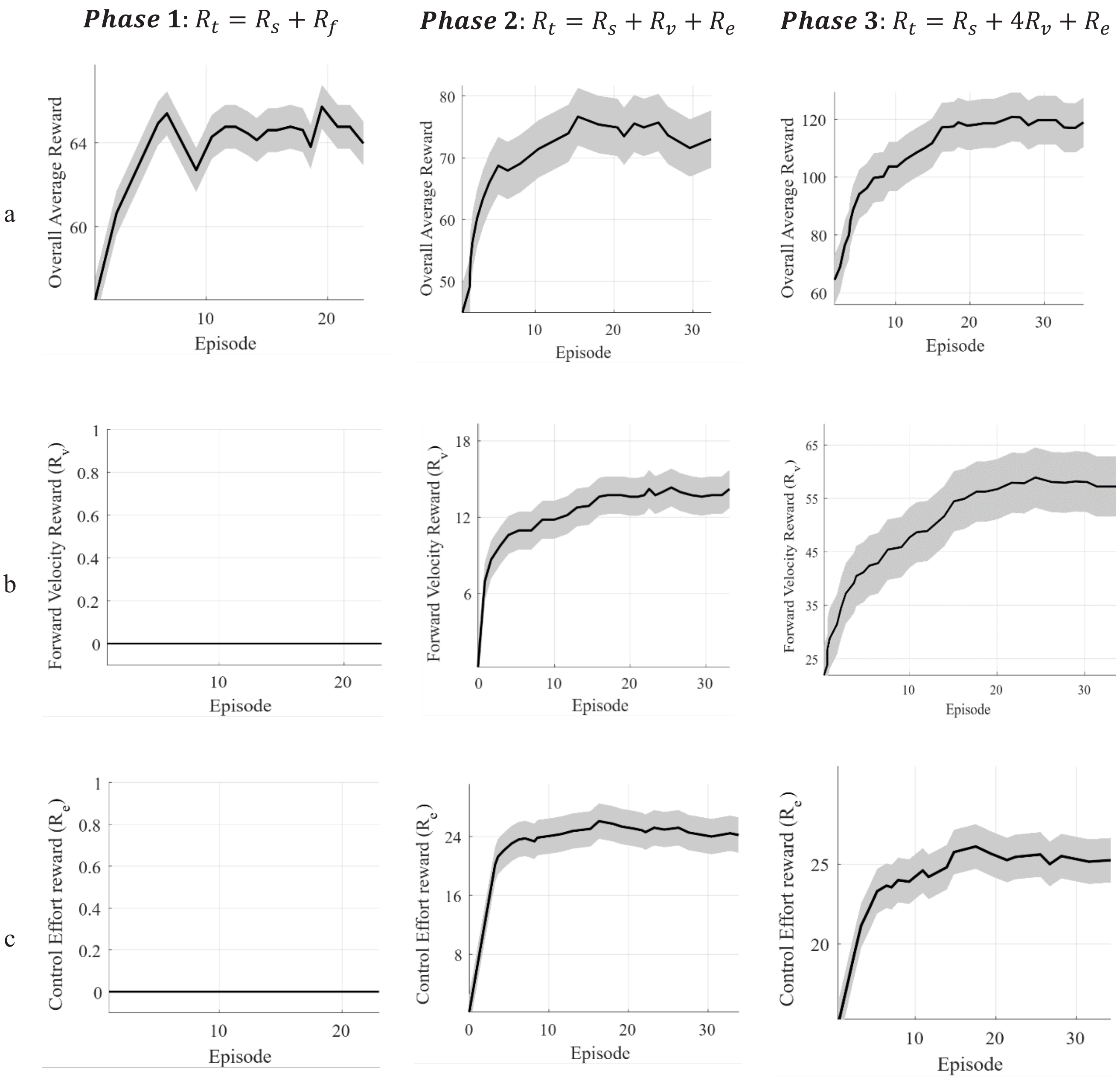


Fig. 6. The rewards during training. Each column consists of three phases as explained in Section II-B. (a) shows overall reward. (b) illustrates the forward velocity regulation reward (R_v in Eq. 1) with desired velocity of v_{ref}=1.5 m/s. (c) presents control effort reward (R_e in Eq. 1).

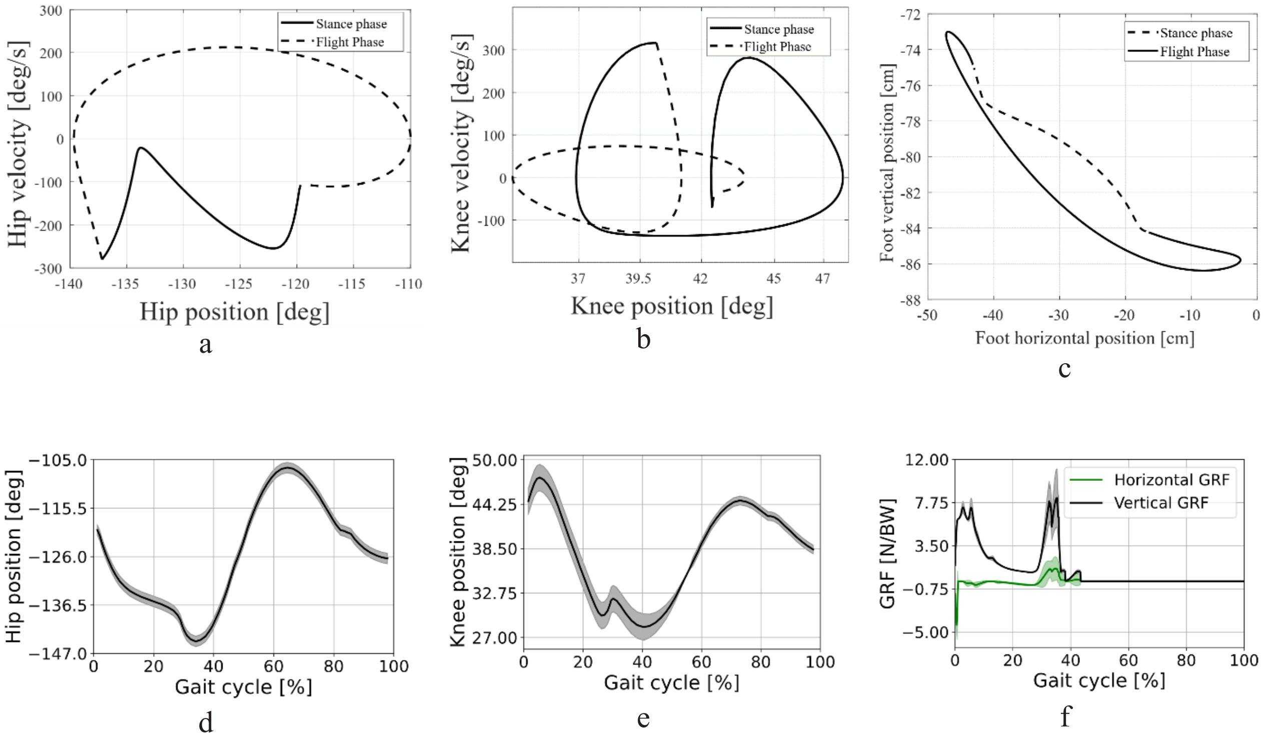


Fig. 7. Evaluation results; the results of the optimal polity. (a-b) present the hip joint and knee joint trajectories in the optimal policy. (c) shows the tip trajectory w.r.t. hip. (d,e) illustrate hip and knee angle variations in a gait stride cycle. (f) show GRF of the simulated single leg based on the soft impact model during walking, which is normalized by body weight (BW). Considering the GRF, it is obvious that all trajectories are started at the impact moment, followed by the stance phase, and then the robot goes to the flight phase about 40% of the gait cycle; i.e., about 40% of the resultant gait cycle is stance phase, and 60% of the resultant gait is the flight phase.

third phases, the control effort-reward contribution weight is the same and non-zero, and the control effort-reward approximately converges to the same level in these two phases. This indicates that the trained policy in the second phase already attains its minimum control effort, such that it cannot be further improved in the third phase, even with different reward function weights.

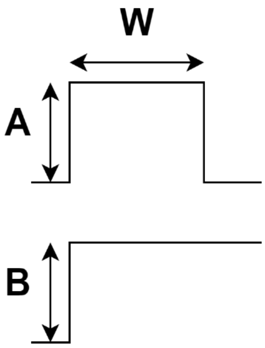
3- 2- Optimal policy evaluation results

The robot's performance after training is presented in Fig. 7, showing the results of the optimal policy. Fig. 7a and Fig. 7b represent hip and knee positions versus their corresponding velocities in state space, respectively; these plots are hip and knee limit cycles. Fig. 7c illustrates the vertical foot position versus the horizontal position, with touch-down and toe-off moments specified. Fig. 7d, Fig. 7e, and Fig. 7f illustrate the average profiles of hip, knee, and ground reaction force (GRF) during one cycle, along with their standard deviations. These sub-figures are plotted using data from 21 stride cycles of the trained model, and the GRFs are normalized by body weight (BW). The ground reaction force is zero during the flight phase (about 40% of the gait cycle) since the leg is not in contact with the ground. As seen in Fig. 7, the results of the trained policy closely resemble locomotion patterns in biology, such as human walking. For instance, similar to human walking, the hip and knee limit cycles in our simulation results (Fig. 7a and Fig. 7b) are self-crossing and non-self-crossing closed curves, where similar behavior can also be observed in human gait (see [38]). Additionally, as shown in Fig. 7e, the ground reaction force in the vertical direction resembles a summation of two Gaussian curves, similar to human walking patterns. Our no-sliding analysis

using the friction cone index[1] indicates that the trained walking policy can maintain a no-sliding condition with a minimum static friction coefficient of $\mu_s \geq 0.8$, ensuring a high level of robustness for mechanical stability.

[1] The friction cone analysis is defined to calculate the minimum friction coefficient between the foot and ground for no sliding condition in legged locomotion [39]. It is computed based on vertical and horizontal ground reaction forces presented in Fig. 6e.

Table 3. Impact of vanishing and non-vanishing disturbances on reward. The table shows the percentage changes in each reward under the different disturbance conditions, illustrating the system's sensitivity to each disturbance type. In this table, the body weight (BW) is about 69N, and the applied forces are reported as a ratio of the body weight.



	Parameters	R_t	R_v	R_e	*Recovered in*
Vanishing	A = 100% BW W = 0.1 s	-50%	-70%	-70%	10 steps
	A = 60% BW W = 0.4 s	-70%	-95%	-40%	7 steps
Non-Vanishing	B = 14% BW	-40%	-70%	-15%	-

3- 3- Robustness results

In our study, robustness is defined as the robot's ability to maintain stable locomotion in face of unknown external disturbance. To evaluate this, two types of disturbances were analyzed:

vanishing and non-vanishing. These disturbances simulate both transient and sustained external forces in the x direction at the hip joint, each designed to push the system to the edges of stability. By evaluating the effect of these disturbances on reward functions—such as total reward (R_t), velocity reward (R_v), and energy efficiency reward (R_e)—we can observe how the control strategy adapts to changes and preserves stable locomotion. The results, are summarized in Table. 3, show percentage reductions in each reward category based on disturbance parameters, revealing sensitivity patterns and recovery capacities.

Our findings show that while stability is maintained, the energy efficiency and forward velocity are significantly impacted by disturbances. In scenarios with vanishing disturbances—short impulses applied to the system—high-magnitude forces caused moderate to severe reductions in R_t, R_v, and R_e Before the system recovered; i.e., at least 50% reduction in total reward. These transient disturbances highlight the sensitivity of trajectory and velocity tracking to impulse forces but also indicate the controller's capacity to return to stable locomotion within a limited number of steps. The impulse in this case is a square pulse, where two parameters can be adjusted: amplitude and width. We considered two cases. In the first case, the pulse width was fixed at $W = 0.1$ seconds, and the force amplitude (A) was gradually increased to approach approximately the body weight of the robot, at which point instability begins to emerge. In the second case, the force amplitude was held constant at a moderate level ($A = 60\%$ of body weight), while the pulse width (W) was incrementally increased until the robot reached the edge of instability with $W = 0.4$ seconds. Notably, as long as these disturbances vanish, the robot takes approximately 10 steps for the first case and 7 steps for the second case to return to its previous gait pattern, demonstrating the controller's ability to recover and reestablish stable locomotion after transient disturbances.

The non-vanishing disturbance tests further validate the controller's robustness, as the system maintained stable operation with degradation in rewards (40% reduction in total reward) despite a constant force applied over time (see Table. 3). This adaptive response, shown by the consistent stability reward (R_s), suggests that the controller can maintain core stability while managing sustained disturbances with an amplitude of $A = 14\%$ of body weight. These results affirm the controller's robustness, demonstrating that it is well-suited for deployment in real-world scenarios where maintaining stability under unknown dynamic and unanticipated conditions is essential.

4- Discussion and Conclusion

This paper presents a novel, simple-to-implement, and bio-inspired control strategy to generate gait cycles for legged robots. The proposed control method utilizes the leg states (position and velocity) at the contact moment to generate a corrective impulsive actuation, making the robot stable, energy-efficient, robust, and capable of regulated forward velocity. This method is ideal for systems employing semi-active actuation mechanisms.

In DC motors, the start torque is much higher than the nominal torque. Using impulsive controllers, such as our suggested controller, allows us to maximize this feature and significantly reduce motor size. Consequently, this reduces the robot's total weight, energy consumption, battery weight, and cost.

Reinforcement learning methods are based on heuristic search, which might lead to falls and severe scenarios for robotic systems. Given the high cost of robotic systems, especially legged robots, it is not safe to run reinforcement learning methods directly on such systems. However, safe reinforcement learning approaches can minimize these risks. One such method is reward shaping, as employed in this paper. Our proposed reward shaping can also be fine-tuned to minimize robot failure scenarios and improve learning safety.

4- 1- Future work

Our next steps are: (1) extend the simulations of our general method to multi-legged systems (e.g., bipedal robots, quadruped robots, etc.), (2) implement the proposed controller and its training strategy in practical applications, and (3) compare the proposed controller with other existing controllers in the literature. These steps will help us further validate and refine our approach, ensuring its effectiveness and robustness across different robotic platforms.

References

[1] P. Biswal and P. K. Mohanty, "Development of quadruped walking robots: A review," Ain Shams Engineering Journal, vol. 12, no. 2, pp. 2017–2031, 2021.

[2] Y. Farid and F. Ruggiero, "Finite-time disturbance reconstruction and robust fractional-order controller design for hybrid port-hamiltonian dynamics of biped robots," Robotics and Autonomous Systems, vol. 144, p. 103836, 2021.

[3] P. A. Bhounsule, J. Cortell, and A. Ruina, "Design and control of ranger: an energy-efficient, dynamic walking robot," in Adaptive Mobile Robotics. World Scientific, 2012, pp. 441–448.

[4] C. Semini, V. Barasuol, T. Boaventura, M. Frigerio, M. Focchi, D. G. Caldwell, and J. Buchli, "Towards versatile legged robots through active impedance control," The International Journal of Robotics Research, vol. 34, no. 7, pp. 1003–1020, 2015.

[5] B. Vanderborght, B. Verrelst, R. Van Ham, M. Van Damme, D. Lefeber, B. M. Y. Duran, and P. Beyl, "Exploiting natural dynamics to reduce energy consumption by controlling the compliance of soft actuators," The International Journal of Robotics Research, vol. 25, no. 4, pp. 343–358, 2006.

[6] J. Zhang, F. Gao, X. Han, X. Chen, and X. Han, "Trot gait design and cpg method for a quadruped robot," Journal of Bionic Engineering, vol. 11, no. 1, pp. 18–25, 2014.

[7] N. Van der Noot, A. J. Ijspeert, and R. Ronsse, "Bio-inspired controller achieving forward speed modulation with a 3d bipedal walker," The International Journal of Robotics Research, vol. 37, no. 1, pp. 168–196, 2018.

[8] R. Nasiri, M. Khoramshahi, and M. N. Ahmadabadi, "Design of a nonlinear adaptive natural oscillator: Towards natural dynamics exploitation in cyclic tasks," in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2016, pp. 3653–3658.

[9] M. Khoramshahi, R. Nasiri, M. Shushtari, A. J. Ijspeert, and M. N. Ahmadabadi, "Adaptive natural oscillator to exploit natural dynamics for energy efficiency," Robotics and Autonomous Systems, vol. 97, pp. 51–60, 2017.

[10] A. J. Ijspeert, A. Crespi, D. Ryczko, and J.-M. Cabelguen, "From swimming to walking with a salamander robot driven by a spinal cord model," science, vol. 315, no. 5817, pp. 1416–1420, 2007.

[11] H. Lee, E. J. Rouse, and H. I. Krebs, "Summary of human ankle mechanical impedance during walking," IEEE journal of translational engineering in health and medicine, vol. 4, pp. 1–7, 2016.

[12] R. Nasiri, M. Khoramshahi, M. Shushtari, and M. N. Ahmadabadi, "Adaptation in variable parallel compliance: Towards energy efficiency in cyclic tasks," IEEE/ASME Transactions on Mechatronics, vol. 22, no. 2, pp. 1059–1070, 2016.

[13] R. Nasiri, A. Ahmadi, and M. N. Ahmadabadi, "Realization of nonlinear adaptive compliance: Towards energy efficiency in cyclic tasks," in 2019 7th International Conference on Robotics and Mechatronics (ICRoM). IEEE, 2019, pp. 175–180.

[14] H. Geyer and H. Herr, "A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities," IEEE Transactions on neural systems and rehabilitation engineering, vol. 18, no. 3, pp. 263–273, 2010.

[15] A. M. Wilson, J. C. Watson, and G. A. Lichtwark, "A catapult action for rapid limb protraction," Nature, vol. 421, no. 6918, pp. 35–36, 2003.

[16] J. Yang, J. Fung, M. Edamura, R. Blunt, R. Stein, and H. Barbeau, "Hreflex modulation during walking in spastic paretic subjects," Canadian Journal of Neurological Sciences, vol. 18, no. 4, pp. 443–452, 1991.

[17] Z. Miranda, A. Pham, G. Elgbeili, and D. Barthélemy, "H-reflex modulation preceding changes in soleus emg activity during balance perturbation," Experimental brain research, vol. 237, no. 3, pp. 777– 791, 2019.

[18] M. Rayati, R. Nasiri, and M. N. Ahmadabadi, "Improving muscle force distribution model using reflex excitation: Toward a model-based exoskeleton torque optimization approach," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 31, pp. 720– 728, 2022.

[19] M. Meinders, A. Gitter, and J. M. Czerniecki, "The role

of ankle plantar flexor muscle work during walking." Scandinavian journal of rehabilitation medicine, vol. 30, no. 1, pp. 39–46, 1998.

[20] M. Srinivasan and A. Ruina, "Computer optimization of a minimal biped model discovers walking and running," Nature, vol. 439, no. 7072, pp. 72–75, 2006.

[21] R. Nasiri, A. Zare, O. Mohseni, M. J. Yazdanpanah, and M. N. Ahmadabadi, "Concurrent design of controller and passive elements for robots with impulsive actuation systems," Control Engineering Practice, vol. 86, pp. 166–174, 2019.

[22] D. Wahrmann, Y. Wu, F. Sygulla, A.-C. Hildebrandt, R. Wittmann, P. Seiwald, and D. Rixen, "Time-variable, event-based walking control for biped robots," International Journal of Advanced Robotic Systems, vol. 15, no. 2, p. 1729881418768918, 2018.

[23] K. A. Hamed, J. Kim, and A. Pandala, "Quadrupedal locomotion via event-based predictive control and qp-based virtual constraints," IEEE Robotics and Automation Letters, vol. 5, no. 3, pp. 4463–4470, 2020.

[24] J. Lee and J. H. Kim, "A comparative study on the l 1 optimal event-based method for biped walking on rough terrains," IEEE Access, vol. 8, pp. 96 304–96 315, 2020.

[25] Y. Lee, H. Lee, J. Lee, and J. Park, "Toward reactive walking: Control of biped robots exploiting an event-based fsm," IEEE Transactions on Robotics, vol. 38, no. 2, pp. 683–698, 2021.

[26] F. Giardina and F. Iida, "Efficient and stable locomotion for impulse-actuated robots using strictly convex foot shapes," IEEE Transactions on Robotics, vol. 34, no. 3, pp. 674–685, 2018.

[27] S. Mochiyama and T. Hikihara, "Impulsive torque control of biped gait with power packets," Nonlinear Dynamics, vol. 102, no., pp. 951–963, 2020.

[28] K. Zhang and E. Braverman, "Event-triggered impulsive control for nonlinear systems with actuation delays," IEEE Transactions on Automatic Control, pp. 1–1, 2022.

[29] J. Weng, E. Hashemi, and A. Arami, "Natural walking with musculoskeletal models using deep reinforcement learning," IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 4156–4162, 2021.

[30] X. B. Peng, G. Berseth, K. Yin, and M. Van De Panne, "Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning," ACM Transactions on Graphics (TOG), vol. 36, no. 4, pp. 1–13, 2017.

[31] T. Li, N. Lambert, R. Calandra, F. Meier, and A. Rai, "Learning generalizable locomotion skills with hierarchical reinforcement learning," in 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2020, pp. 413–419.

[32] H. M. Clayton, H. C. Schamhardt, M. A. Willemen, J. L. Lanovaz, and G. R. Colborne, "Kinematics and ground reaction forces in horses with superficial digital flexor tendinitis," American journal of veterinary research, vol. 61, no. 2, pp. 191–196, 2000.

[33] N. Ogihara, E. Hirasaki, H. Kumakura, and M. Nakatsukasa, "Ground reaction-force profiles of bipedal walking in bipedally trained Japanese monkeys," Journal of human evolution, vol. 53, no. 3, pp. 302–308, 2007.

[34] H. T. Lin and B. A. Trimmer, "The substrate as a skeleton: ground reaction forces from a soft-bodied legged animal," Journal of Experimental Biology, vol. 213, no. 7, pp. 1133–1142, 2010.

[35] S. W. Lipfert, Kinematic and dynamic similarities between walking and running. Kovaˇc Hamburg, 2010.

[36] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in International Conference on Machine Learning. PMLR, 2018, pp. 1587–1596.

[37] A. D. Laud, Theory and application of reward shaping in reinforcement learning. University of Illinois at Urbana-Champaign, 2004.

[38] R. Nasiri, H. Dinovitzer, and A. Arami, "A unified gait phase estimation and control of exoskeleton using virtual energy regulator (ver)," in 2022 International Conference on Rehabilitation Robotics (ICORR). IEEE, 2022, pp. 1–6.

[39] S. Kajita, K. Kaneko, K. Harada, F. Kanehiro, K. Fujiwara, and H. Hirukawa, "Biped walking on a low friction floor," in 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No. 04CH37566), vol. 4. IEEE, 2004, pp. 3546– 3552.

[40] Zhao, Yongyong, Jinghua Wang, Guohua Cao, Yi Yuan, Xu Yao, and Luqiang Qi. "Intelligent control of multilegged robot smooth motion: a review." IEEE Access 11 (2023): 86645-86685.

[41] Kotha, Swapnil Saha, Nipa Akter, Sarafat Hussain Abhi, Sajal Kumar Das, Md Robiul Islam, Md Firoj Ali, Md Hafiz Ahamed et al. "Next generation legged robot locomotion: A review on control techniques." Heliyon (2024).

[42] Gao, Yong, Wu Wei, Xinmei Wang, Dongliang Wang, Yanjie Li, and Qiuda Yu. "Trajectory tracking of multi-legged robot based on model predictive and sliding mode control." Information Sciences 606 (2022): 489-511.

[43] Morimoto, Jun, and Christopher Atkeson. "Minimax differential dynamic programming: An application to robust biped walking." Advances in neural information processing systems 15 (2002).

[44] Peng, Xue Bin, Erwin Coumans, Tingnan Zhang, Tsang-Wei Lee, Jie Tan, and Sergey Levine. "Learning agile robotic locomotion skills by imitating animals." arXiv preprint arXiv:2004.00784 (2020). Alexander, R. McNeill. Principles of animal locomotion. Princeton university press, 2003.

[45] Fu, Zipeng, Ashish Kumar, Jitendra Malik, and Deepak Pathak. "Minimizing energy consumption leads to the emergence of gaits in legged robots." arXiv preprint

arXiv:2111.01674 (2021).

[46] Hutter M, Gehring C, Jud D, Lauber A, Bellicoso CD, Tsounis V, Hwangbo J, Bodie K, Fankhauser P, Bloesch M, Diethelm R. Anymal-a highly mobile and dynamic quadrupedal robot. In2016 IEEE/RSJ international conference on intelligent robots and systems (IROS) 2016 Oct 9 (pp. 38-44). IEEE.

Appendix A: Derivation of Dynamic Equations for Single-Legged Hopping Model

This appendix provides a detailed derivation of the dynamic equations for the planar single-legged robot. The dynamics of this robotic model is described as:

$$D(\Theta)\ddot{\Theta} + C\big(\Theta,\dot{\Theta}\big) + G(\Theta) = \boldsymbol{B}^T F_{ag} + \boldsymbol{E}^T F_{pg} + F_c(q) + F_b(v),\ \ F_{ag} = \big[F_x, F_y\big]^T$$

where $\Theta = [q, x, y]^T = \big[\theta_{hip}, \theta_{knee}, x, y\big]^T \in \mathbb{R}^4$ denotes the generalized coordinates of the robot with 4 degrees of freedom. $\dot{\Theta} = \big[\omega_{hip}, \omega_{knee}, v_x, v_y\big]^T \in \mathbb{R}^4$ is the derivative of the generalized coordinates. F_x and F_y are actuation forces applied to the robotic model from the ground in x and y directions. In addition, $D \in \mathbb{R}^{4\times4}$ denotes the inertia matrix, $C \in \mathbb{R}^4$ denotes the Coriolis and centrifugal forces vector, and $G \in \mathbb{R}^4$ denotes the generalized gravity vector. The term $F_{ext} \in \mathbb{R}^2$ denotes the GRFs acting on the robot's contacting feet and $\boldsymbol{B} = \boldsymbol{E} \in \mathbb{R}^{2\times4}$ denotes the Jacobian of the associated contact frame. $F_c(q)$ and $F_b(v)$ The detailed derivations of the dynamical equations are presented as follow.

$$\boldsymbol{B} = \boldsymbol{E} = \begin{bmatrix} -l_1\sin(\theta_{hip}) - l_2\sin(\theta_{hip}+\theta_{knee}) & -l_2\sin(\theta_{hip}+\theta_{knee}) & 1 & 0 \\ l_1\cos(\theta_{hip}) + l_2\cos(\theta_{hip}+\theta_{knee}) & l_2\cos(\theta_{hip}+\theta_{knee}) & 0 & 1 \end{bmatrix}$$

$$D_{11} = \frac{m_1(l_1^2+d^2)}{12} + I_2 + \frac{l_1^2 m_1}{4} + l_1^2 m_2 + \frac{l_2^2 m_2}{4} + l_1 l_2 m_2\cos(\theta_{knee})\ \ , D_{22} = \frac{m_2 l_2^2}{4} + \frac{m_2(l_2^2+d^2)}{12}$$

$$D_{33} = m_1 + m_2\ , D_{44} = m_1 + m_2\ \ , D_{12} = D_{21} = \frac{m_2(l_2^2+d^2)}{12} + \frac{m_2 l_2^2}{4} + \frac{l_1 m_2\cos(\theta_{knee})\, l_2}{2}$$

$$D_{13} = D_{31} = -\frac{m_2 l_2\sin\big(\theta_{hip}+\theta_{knee}\big)}{2} + m_2 l_1\sin(q_1)\ - \frac{l_1 m_1\sin\big(\theta_{hip}\big)}{2}$$

$$D_{14} = D_{41} = \frac{l_2 m_2\cos\big(\theta_{hip}+\theta_{knee}\big)}{2} + m_2 l_1\cos(q_1)\ +\ l_1 m_1\frac{\cos\big(\theta_{hip}\big)}{2}$$

$$D_{24} = D_{42} = \frac{l_2 m_2\cos\big(\theta_{hip}+\theta_{knee}\big)}{2}\ \ ,\qquad D_{34} = D_{43} = 0$$

$$C_1 = -\,\omega_{knee}^2\frac{l_1 l_2 m_2\sin(\theta_{knee})}{2} - \omega_{hip}\omega_{knee} l_1 l_2 m_2\sin(\theta_{knee})\ ,\qquad C_2 = \omega_{hip}^2 l_2 m_2 l_1 sin(\theta_{knee})$$

$$C_3 = -\omega_{hip}^2 m_2 l_1\cos(q1)\ + \frac{m_2 l_2\cos(q1\ +\ q2)\big(\omega_{hip}\ + \omega_{knee}\big)^2}{2} - \frac{\omega_{hip}^2 l_1 m_1\cos\big(\theta_{hip}\big)}{2}$$

$$C_4 = \omega_{hip}^2 m_2 l_1\sin(q1) + \frac{m_2 l_2\sin(q1\ +\ q2)\big(\omega_{hip}\ + \omega_{knee}\big)^2}{2} - \frac{\omega_{hip}^2 l_1 m_1\sin\big(\theta_{hip}\big)}{2}$$

$$G_1 = \frac{g l_2 m_2\cos\big(\theta_{hip}+\theta_{knee}\big)}{2} + \frac{g l_1 m_1\cos\big(\theta_{hip}\big)}{2} +\ g l_1 m_2\cos(\theta_{hip})$$

$$G_2 = \frac{l_2 m_2 g\cos\big(\theta_{hip}+\theta_{knee}\big)}{2}\ \ ,\qquad G_3 = 0\ \ ,\qquad G_4 = g(m_1 + m_2)$$

HOW TO CITE THIS ARTICLE

B. S. Mortazavi, R. Nasiri, M. Nili Ahmadabadi, A Simplified Event-based Impulsive Control Approach for Stable, Efficient, and Robust Locomotion Using Deep Reinforcement Learning, AUT J. Model. Simul., 56(2) (2024) 185-198.

DOI: 10.22060/miscj.2025.23344.5368