Adaptive Gait Control for Quadruped Robots on Varied Slopes via ARS Algorithm

Van-Truong Nguyen
School of Mechanical and Automotive Engineering, Hanoi University of Industry, Vietnam
nguyenvantruong@haui.edu.vn (corresponding author)

Ngoc-Quyet Nguyen
School of Mechanical and Automotive Engineering, Hanoi University of Industry, Vietnam
quyet.nguyen.official@gmail.com

Thanh-Lam Bui
School of Mechanical and Automotive Engineering, Hanoi University of Industry, Vietnam
buithanhlam@haui.edu.vn

Received: 26 November 2024 | Revised: 29 December 2024 and 12 January 2025 | Accepted: 18 January 2025

 DOI: https://doi.org/10.48084/etasr.9692

ABSTRACT

Ensuring stable walking for quadruped robots on unknown slopes is a critical challenge in robotic navigation. This study introduces a novel gait planning algorithm that leverages data from an Inertial Measurement Unit (IMU) for terrain slope estimation, offering a cost-effective alternative to visual sensors. The proposed approach integrates a trot gait with an elliptical foot trajectory, enabling efficient movement across varied slopes. Using the Augmented Random Search (ARS) algorithm, we fine-tune the elliptical trajectory parameters to achieve precise and adaptive foot placements. Additionally, the robot dynamically adjusts its posture in real time to maintain stability by aligning with desired joint angles during slope traversal. Simulation results validate the effectiveness of the proposed algorithm, demonstrating its ability to ensure stable and adaptive locomotion on slopes of up to 11 degrees. This work highlights the feasibility of using low-cost hardware and advanced algorithms to address complex terrain navigation challenges.

Keywords-reinforcement learning; Augmented Random Search (ARS); artificial intelligence; quadruped robot; robot stability

I. INTRODUCTION

The development of quadruped robots has advanced robotic capabilities, enabling the navigation of complex terrains where traditional wheeled robots fall short [1, 2]. Their adaptive leg designs make them valuable in education, healthcare, service industries, industrial applications, and military operations [3-6]. However, gaps remain in matching their biological counterparts, especially on uneven terrains [7-10]. This study introduces "Spot Dog," a cost-effective quadruped robot (see Figure 1) that uses servo motors instead of the BLDC motors used in modern robots [11], thus overcoming the challenge of less precise torque control [12]. Nevertheless, Spot Dog demonstrates effective operation on rough terrains. Previous research has focused on static gaits [13], with Professor McGhee's team pioneering stable static gait walking on flat and irregular surfaces [14, 15]. Additionally, layered motion controllers have been developed for foothold selection on challenging terrains, though real-time performance and parameter optimization remain challenges [16].

Inspired by biological systems, the Central Pattern Generator (CPG) approach has been employed to improve the stability and adaptability of robot motion [17]. For example, authors in [18] developed a CPG-based controller for omnidirectional motion, whereas authors in [19] integrated CPG with kinematic and dynamic models to design foot trajectories for challenging terrains. However, these methods often rely on simplistic feedback mechanisms, limiting the full exploitation of CPG stability properties. Motion planning and trajectory optimization methods for robots have also been extensively explored [20-32]. Especially for quadruped robots, authors in [20] proposed a tree search for body path and footstep planning in the HyQ robot, and authors in [21, 22] introduced a ZMP-based planner for dynamic gait transitions in ANYmal. Model-predictive control (MPC) has been applied to centroidal models for Mini-cheetah in [23] and in [24, 25], optimizing base trajectories and reaction forces while simplifying whole-body control. Despite their robustness, these methods often rely on hand-tuned parameters, require significant computational resources, and face challenges in adapting to real-world environments.

This research introduces Spot Dog, a cost-effective quadruped robot designed to navigate challenging terrains using servo motors. Our approach leverages the Augmented Random Search (ARS) algorithm [26] to optimize a linear control policy, reducing computational requirements while maintaining robust performance. Spot Dog employs a trot gait [27] with a semi-elliptical endpoint trajectory, which allows for smooth and stable locomotion. By combining low-cost hardware with advanced algorithms, our work demonstrates a practical and efficient solution for real-world applications. Spot Dog's successful performance across various terrains validates the feasibility of using servo motors for quadruped robots, providing an accessible platform for further research and development.

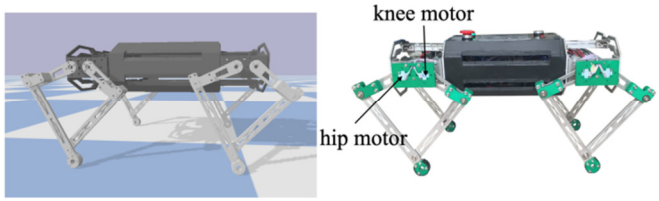

Fig. 1. Spot Dog robot.

II. KINEMATIC ANALYSIS

Kinematic analysis is critical to controlling the Spot Dog robot. Forward kinematics calculates leg positions from joint angles, whereas inverse kinematics determines joint angles for desired leg positions, improving precision and performance. This study provides the foundational methods for both.

The leg mechanism is modeled as shown in Figure 2, with the coordinate origin located at the hip joint A(O). The joints are labeled sequentially as A, B, C, E, and F. The lengths of the segments AF, AB, EF, BC, CE, and CD are l, l_1, l_1, l_2, l_2, and l_3, respectively. The angles between the joints and the x-axis are denoted $\theta_1, \theta_2, \theta_3$, and θ_4. Additionally, the coordinates of the endpoint of the leg D are represented by (x, y).

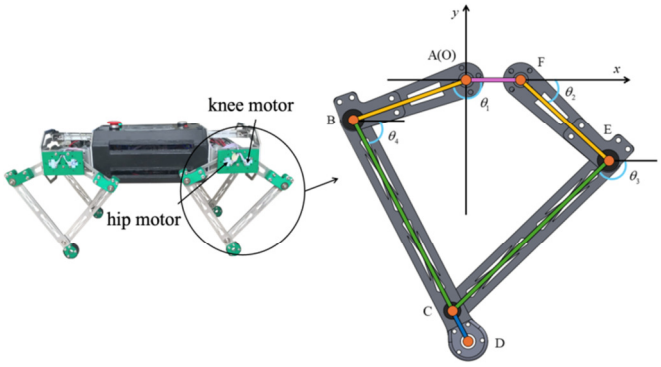

Fig. 2. Five-bar leg mechanism.

A. Forward Kinematics

Let's temporarily ignore the length of segment CD. By projecting the lengths of the remaining segments onto the x and y axes, we obtain the following system of equations:

$$\begin{cases} l + l_1\cos\theta_2 + l_2\cos\theta_3 = l_1\cos\theta_1 + l_2\cos\theta_4 \\ l_1\sin\theta_2 + l_2\sin\theta_3 = l_1\sin\theta_1 + l_2\sin\theta_4 \end{cases} \quad (1)$$

We know the exact values of l, l_1, and l_2. Additionally, the angles θ_1 and θ_2 are active angles, so the unknown variables are θ_3 and θ_4. By moving the unknown variables to the left side, (1) becomes:

$$\begin{cases} l_2\cos\theta_3 - l_2\cos\theta_4 = l_1\cos\theta_1 - l_1\cos\theta_2 - l \\ l_2\sin\theta_3 - l_2\sin\theta_4 = l_1\sin\theta_1 - l_1\sin\theta_2 \end{cases} \quad (2)$$

Let:

$$\begin{cases} u = l_1\cos\theta_1 - l_1\cos\theta_2 - l \\ t = l_1\sin\theta_1 - l_1\sin\theta_2 \end{cases} \quad (3)$$

Equation (2) becomes:

$$\begin{cases} l_2\cos\theta_3 - l_2\cos\theta_4 = u \\ l_2\sin\theta_3 - l_2\sin\theta_4 = t \end{cases} \quad (4)$$

By solving the system of (3) using MATLAB, we can calculate the values of the joint angles θ_3 and θ_4:

$$\begin{cases} \theta_3 = 2\arctan\left(\dfrac{2tl_2 \pm \sqrt{-\left(u^2+t^2\right)\left(u^2-4l_2^2+t^2\right)}}{u^2+2ul_2+t^2}\right) \\ \theta_4 = -2\arctan\left(\dfrac{2tl_2 \pm \sqrt{-\left(u^2+t^2\right)\left(u^2-4l_2^2+t^2\right)}}{u^2-2ul_2+t^2}\right) \end{cases} \quad (5)$$

By combining this with the length of segment CD, we can calculate the coordinates of the leg's endpoint D as follows:

$$\begin{cases} x = l_1\cos\theta_1 + \left(l_2+l_3\right)\cos\theta_4 \\ y = l_1\sin\theta_1 + \left(l_2+l_3\right)\sin\theta_4 \end{cases} \quad (6)$$

B. Inverse Kinematics

For the inverse kinematics problem, we can determine the relationship between point D with coordinates (x, y), the final position of the robot's leg, and the motor control angles θ_1 and θ_2, thus achieving the goal of controlling the endpoint position. First, we divide the five-bar linkage mechanism into two branches: branch 1 with segments AB and BD, having lengths l_1 and $l_t = l_2 + l_3$, respectively, to find the angle θ_1; branch 2 with segments EF and CE, having lengths l_1 and l_2, respectively, to find the angle θ_2.

For branch 1, let:

$$\begin{cases} a = 2xl_t \\ b = 2yl_t \\ c = l_1^2 - l_t^2 - x^2 - y^2 \end{cases} \quad (7)$$

We can find the angle θ_4 as follows:

$$\theta_4 = \arctan 2(y, x) + \arccos \frac{-c}{\sqrt{a^2 + b^2}} \quad (8)$$

After obtaining the angle θ_4, the angle θ_1 is determined as follows:

$$\theta_1 = \arctan 2(y - l_t \sin\theta_4, x - l_t \cos\theta_4) \quad (9)$$

For branch 2, we can calculate the coordinates of point C using the angle θ_4:

$$\begin{cases} x_C = x - l_3 \cos\theta_4 \\ y_C = y - l_3 \sin\theta_4 \end{cases} \quad (10)$$

Now, we need to subtract l from x_C because point F is at a distance l from the origin A(O). Then, we calculate the angle θ_2 in a similar manner as described above for θ_1.

$$\begin{cases} x_n = x_C - l \\ y_n = y_C \end{cases} \quad (11)$$

Let:

$$\begin{cases} m = 2x_n l_2 \\ n = 2y_n l_2 \\ p = l_1^2 - l_2^2 - x_n^2 - y_n^2 \end{cases} \quad (12)$$

Next, we calculate the angles θ_3 and θ_2:

$$\theta_3 = \arctan 2(x_n, y_n) - \arccos \frac{-p}{\sqrt{m^2 + n^2}} \quad (13)$$

$$\theta_2 = \arctan 2(y_n - l_2 \sin\theta_3, x_n - l_2 \cos\theta_3) \quad (14)$$

III. REINFORCEMENT LEARNING FRAMEWORK

A. Observation Space

The quadruped robot's observation space includes its base orientations and terrain slope parameters, which are essential for determining its state and making action decisions. Equipped with an IMU sensor, the robot measures base angles (roll, pitch, yaw) and terrain slope angles (roll, pitch) to operate effectively on various terrains. Overall, the observation space is defined as an 11-dimensional vector:

$$s_t = [\Phi_{t-2}, \Phi_{t-1}, \Phi_t, \gamma_t, \sigma_t] \quad (15)$$

where $\Phi_t \in \mathbb{R}^3$ represents the base orientations of the robot, γ_t and σ_t denote the roll and pitch angles of the sloped terrain at time step t. Observations of the robot's base orientations at time steps $t-2$ and $t-1$ are also crucial for enhancing the robot's performance.

B. Action Space

The robot's action space is a 12-dimensional vector that adjusts each leg's semi-elliptical trajectory for flexible and stable movement. It includes step length parameters that affect speed and stability, as well as deviations along the x and z axes to enable direction changes and terrain adaptation.

$$a_t = [l_t, x_t, z_t] \quad (16)$$

where $l_t, x_t, z_t \in \mathbb{R}^4$, l_t represents the step lengths, x_t and z_t are additional deviations along the x and z axes of the semi-ellipse. A detailed description of our semi-elliptical trajectory generator can be found in [9].

C. Reward Function

The reward function is designed to evaluate the robot's performance in terms of movement and stability. Specifically, our reward function is formulated as follows:

$$r = G_{w1}(e_R) + G_{w2}(e_P) + G_{w3}(e_Y) + G_{w4}(e_H) + W\Delta x \quad (17)$$

where e_R, e_P, e_Y and e_H represent the deviations in roll, pitch, yaw, and height angles, respectively. Δx is the distance the robot moves along the x-axis. The function G denotes a Gaussian kernel function mapping $G: R \to [0,1]$. The reward function balances multiple aspects of the robot's performance by minimizing the deviations in roll, pitch, yaw, and height $G_{w1}(e_R)$, $G_{w2}(e_p)$, $G_{w3}(e_Y)$, and $G_{w4}(e_H)$, respectively, to maintain stability and proper orientation on various terrains. These deviations are penalized using a Gaussian kernel function $G_{w_j} = \exp(-w_j x^2)$, where $w_j > 0$, that controls the curve's width, ensuring that larger errors receive larger penalties and promoting precise, controlled movements. Additionally, the reward includes a term Δx that encourages forward movement along the x-axis, promoting efficient locomotion. The weight W adjusts the emphasis between forward movement and stability/orientation control. Overall, the reward function achieves a balance between stability, orientation, and efficient locomotion by penalizing deviations and rewarding forward progress. The use of Gaussian penalties enhances precision, guiding the robot to optimize its performance across various terrains and to navigate effectively in complex environments.

D. Policy

The robot's policy π is represented as a matrix that maps an 11-dimensional state vector to a 12-dimensional action array. Specifically, the policy matrix translates the robot's current state, described by an 11-dimensional vector, into actions within a 12-dimensional space. Mathematically, the policy π can be expressed as:

$$a_t = \pi(s_t) = Ws_t \quad (18)$$

where $W \in \mathbb{R}^{12\times11}$ is the policy matrix, $s_t \in \mathbb{R}^{11}$ is the state vector at time t, and $a_t \in \mathbb{R}^{12}$ is the action vector at time t. The training aims to optimize the policy matrix, enabling actions that maximize the robot's reward and performance under various conditions.

E. Training Algorithm

The training algorithm used in this study is ARS, specifically the V-1t version [27]. ARS optimizes the policy parameters $\theta \in \mathbb{R}^{12\times11}$ by following the gradient of the expected reward function. Unlike algorithms such as proximal policy optimization and trust region policy optimization, which use likelihood ratio methods, ARS employs finite differences to estimate the gradient. The process begins with random initialization of the policy parameters θ. Perturbations sampled from a normal distribution are applied to θ to create perturbed policies. These policies are then evaluated through rollouts, and the resulting rewards are used to update θ. By iteratively refining the parameters based on these rewards, ARS V-1t seeks to find the optimal policy parameters that maximize the expected reward, thereby enhancing the robot's performance. The pseudocode for ARS (version V-1t) is as follows:

Algorithm 1 Augmented Random Search (ARS) V-1t

Initialize $\theta \in R^{12\times11}$ randomly
While ending condition not satisfied do
  Sample perturbations $\delta_1, \delta_2, \ldots, \delta_N$ from $N(0, I)$
  For $i = 1, 2, \ldots, N$ do
    $\theta_{plus} = \theta + \nu\delta_i$, $\theta_{minus} = \theta - \nu\delta_i$
    $r_{plus} = \text{Rollout}(\theta_{plus})$, $r_{minus} = \text{Rollout}(\theta_{minus})$
  End for
  Sort the perturbations δ_i based on $\max(r_{plus}, r_{minus})$
  Compute the update $\Delta\theta$ using the top b rewards
  $\Delta\theta = \frac{1}{b\sigma_R}\sum_{i=1}^{b}(r_{plus,i} - r_{minus,i})\delta_i$
  Update the policy
  $\theta = \theta + \alpha\Delta\theta$
End while

In this pseudocode, α is the learning rate, ν is the standard deviation of the perturbations, r_{plus} and r_{minus} are the rewards obtained from the perturbed policies, and σ_R is the standard deviation. By following this procedure, ARS effectively searches for the optimal policy parameters that maximize the robot's performance across various environmental conditions.

IV. SIMULATION AND RESULTS

The simulation environment includes terrains with slopes up to 11 degrees. The ARS algorithm is configured with a learning rate $\alpha = 0.02$, noise $\nu = 0.03$, 200 training episodes, each consisting of 500 simulation time steps. These parameters were chosen to balance exploration and stability, allowing efficient policy updates without overshooting or underfitting. The learning rate controls the step size for policy updates, whereas noise ensures sufficient exploration of gait adjustments. The number of episodes provides the algorithm with adequate training data to effectively generalize across varied slopes. Simulations were run on a computer with an Intel i7 processor and 16GB RAM, using Python and the PyBullet simulation environment, with the robot integrated into the Gym environment. The training procedure follows the ARS algorithm, starting with the initialization of a policy matrix. Perturbations sampled from a normal distribution are applied to the policy parameters to explore the parameter space. Training is conducted in two stages: initial training on simpler terrains with slopes ranging from 5 to 7 degrees for approximately 40 episodes, followed by more challenging terrains with slopes of 9 and 11 degrees. After each episode, the updated policy is tested on all terrain slopes (5, 7, 9, and 11 degrees) to calculate the average reward, which evaluates the robot's stability, posture adjustments, and forward movement. This staged training approach allows the robot to progressively adapt to increasing complexity. The entire training process, consisting of 200 episodes with 500-time steps per episode, was completed within 11 to 12 hours.

The graph in Figure 3 depicts the average reward per episode during training in the simulated environment. In the initial 40 episodes, where the robot was trained on slopes of 5 to 7 degrees, the reward values remained relatively stable. This phase corresponds to the early stage of learning, where the robot explores actions with limited immediate performance improvement. After gaining sufficient experience, the robot began training on steeper slopes (9 to 11 degrees), which resulted in a consistent upward trend in rewards. This steady improvement indicates that the ARS algorithm effectively adjusted the robot's gait parameters to adapt to more challenging terrains. The trend not only demonstrates reliable performance enhancement, but also the robustness and stability of the learning process. These results validate the efficacy of the reinforcement learning approach in enabling the robot to learn adaptive and efficient gait patterns.

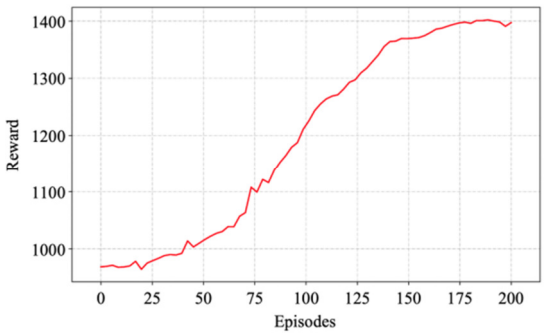

Fig. 3. Average reward during robot training.

Figure 4 illustrates the roll and pitch angles of the robot's body as it traverses flat terrain with a 0-degree slope. The blue line represents the raw IMU sensor data, the orange line shows the estimated slope angles calculated using the ARS-based policy, and the green line indicates the actual angles. The close alignment between the estimated angles (orange) and the actual angles (green) indicates the accuracy of the slope estimation mechanism. The minimal deviations in the roll and pitch angles (blue) further validate the robot's ability to maintain a stable posture during locomotion on flat terrain. This confirms the effectiveness of the gait planning strategy in achieving balance and stability under minimal external disturbances.

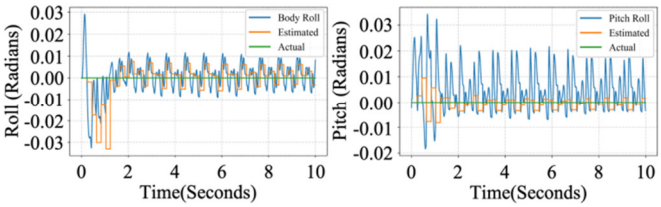

Fig. 4. Roll and pitch angle change on a 0-degree slope.

Figures 5 and 6 compare the roll and pitch angle changes on slopes of 5 and 11 degrees. The blue line represents the ARS algorithm, the orange line represents the PID algorithm [28, 29], and the green line represents the actual angles. In Figure 5, the roll angle shows minimal fluctuations and closely follows the actual values, demonstrating stability on gentler slopes. Figure 6 highlights three phases: transitioning to the slope, steady climbing, and passing the crest. The ARS algorithm consistently outperforms the PID controller, maintaining better stability and enabling successful navigation even on steeper slopes. These results confirm the feasibility and effectiveness of the ARS approach.

Figures 7 and 8 show the distance traveled by the robot's center of gravity over 10 seconds on slopes of 5 and 11 degrees. While steeper slopes reduce the robot's speed, the trajectory remains stable and consistently forward-moving, showcasing the ARS algorithm's ability to maintain control and direction.

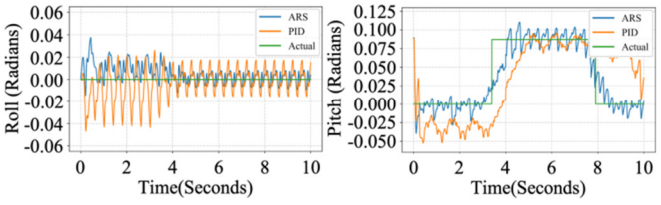

Fig. 5. Roll and pitch angle change on a 5-degree slope.

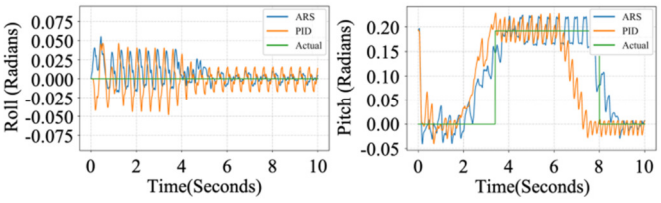

Fig. 6. Roll and pitch angle change on an 11-degree slope.

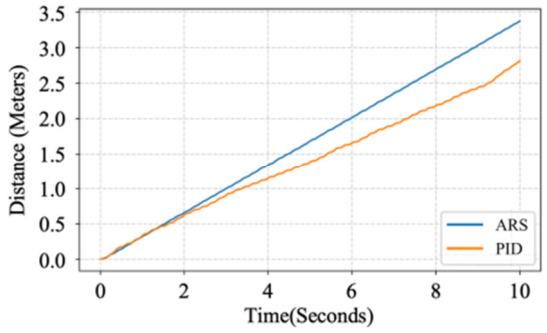

Fig. 7. Distance the center of gravity has moved on a 5-degree slope.

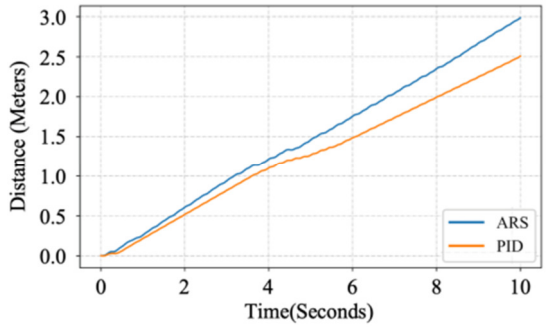

Fig. 8. Distance the center of gravity has moved on an 11-degree slope.

V. CONCLUSION

In this study, we developed a cost-effective quadruped robot using DC motors, making advanced-legged robotics research more accessible compared to traditional high-cost solutions employing BLDC motors or proprietary designs. A reinforcement learning-based control algorithm, specifically the Augmented Random Search (ARS) algorithm, was implemented and evaluated for navigating terrains with slopes up to 11 degrees. The results demonstrate that even with low-cost hardware, the robot achieved impressive performance, outperforming traditional PID controllers in stability and adaptability on challenging slopes.

A key novelty of this work lies in the integration of ARS with a reward function tailored to account for Gaussian errors in roll, pitch, yaw, height, and distance traveled, enabling efficient learning of adaptive gait patterns. During training, the average reward steadily increased by over 70%, highlighting the effectiveness of the ARS algorithm in improving the robot's stability and adaptability. On a 5-degree slope, the roll and pitch angle deviations remained within 2 degrees, while on an 11-degree slope, the deviations were maintained within 3.5 degrees. Additionally, the robot achieved a forward-moving trajectory, covering a stable and straight path over a 10-second test period, even on the steepest terrains. These quantitative results further validate the feasibility of using DC motor-driven quadruped robots in real-world scenarios without sacrificing performance. Compared to existing legged robotics research, this study bridges the gap between affordability and capability, providing a practical platform for broader experimentation and application. Future work will focus on enhancing the robot's adaptability to more complex terrains, refining the reward functions and hyperparameters, exploring the integration of

additional sensors to further improve its stability and efficiency, and experimenting with other control algorithms to evaluate and compare their effectiveness.

ACKNOWLEDGMENT

This work is financially supported by the Hanoi University of Industry (Grant No: 44-2024-RD/HĐ-ĐHCN).

REFERENCES

[1] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, Oct. 2020, Art. no. eabc5986, https://doi.org/10.1126/scirobotics.abc5986.

[2] H. Chai *et al.*, "A survey of the development of quadruped robots: Joint configuration, dynamic locomotion control method and mobile manipulation approach," *Biomimetic Intelligence and Robotics*, vol. 2, no. 1, Mar. 2022, Art. no. 100029, https://doi.org/10.1016/j.birob.2021.100029.

[3] P. Biswal and P. K. Mohanty, "Development of quadruped walking robots: A review," *Ain Shams Engineering Journal*, vol. 12, no. 2, pp. 2017–2031, Jun. 2021, https://doi.org/10.1016/j.asej.2020.11.005.

[4] D. M. Duc, T. X. Tuy, and P. D. Phuoc, "A Study on the Response of the Rehabilitation Lower Device using Sliding Mode Controller," *Engineering, Technology & Applied Science Research*, vol. 11, no. 4, pp. 7446–7451, Aug. 2021, https://doi.org/10.48084/etasr.4312.

[5] V.-T. Tran, B.-S. Nguyen, T. Vu, and N.-T. Bui, "Trajectory Tracking Control of Pneumatic Cylinder-Actuated Lower Limb Robot for a Gait Training System," *Engineering, Technology & Applied Science Research*, vol. 14, no. 4, pp. 15367–15372, Aug. 2024, https://doi.org/10.48084/etasr.7733.

[6] N. Kosmyna, E. Hauptmann, and Y. Hmaidan, "A Brain-Controlled Quadruped Robot: A Proof-of-Concept Demonstration," *Sensors*, vol. 24, no. 1, Jan. 2024, Art. no. 80, https://doi.org/10.3390/s24010080.

[7] E. Garcia, M. A. Jimenez, P. G. De Santos, and M. Armada, "The evolution of robotics research," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 90–103, Mar. 2007, https://doi.org/10.1109/MRA.2007.339608.

[8] S. Coros, A. Karpathy, B. Jones, L. Reveret, and M. van de Panne, "Locomotion skills for simulated quadrupeds," *ACM Transactions on Graphics*, vol. 30, no. 4, Jul. 2011, Art. no. 59, https://doi.org/10.1145/2010324.1964954.

[9] C. Ding, L. Zhou, Y. Li, and X. Rong, "A Novel Dynamic Locomotion Control Method for Quadruped Robots Running on Rough Terrains," *IEEE Access*, vol. 8, pp. 150435–150446, 2020, https://doi.org/10.1109/ACCESS.2020.3016312.

[10] V.-G. Loc *et al.*, "Sensing and gait planning of quadruped walking and climbing robot for traversing in complex environment," *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 666–675, May 2010, https://doi.org/10.1016/j.robot.2009.11.007.

[11] D. B. Minh, V. D. Quoc, and P. N. Huy, "Efficiency Improvement of Permanent Magnet BLDC Motors for Electric Vehicles," *Engineering, Technology & Applied Science Research*, vol. 11, no. 5, pp. 7615–7618, Oct. 2021, https://doi.org/10.48084/etasr.4367.

[12] J. Estremera and K. J. Waldron, "Thrust Control, Stabilization and Energetics of a Quadruped Running Robot," *The International Journal of Robotics Research*, vol. 27, no. 10, pp. 1135–1151, Oct. 2008, https://doi.org/10.1177/0278364908097063.

[13] Q. Hao, Z. Wang, J. Wang, and G. Chen, "Stability-Guaranteed and High Terrain Adaptability Static Gait for Quadruped Robots," *Sensors*, vol. 20, no. 17, Sep. 2020, Art. no. 4911, https://doi.org/10.3390/s20174911.

[14] R. B. McGhee and A. A. Frank, "On the stability properties of quadruped creeping gaits," *Mathematical Biosciences*, vol. 3, pp. 331–351, Aug. 1968, https://doi.org/10.1016/0025-5564(68)90090-4.

[15] H. Hwang and Y. Youm, "Steady Crawl Gait Generation Algorithm for Quadruped Robots," *Advanced Robotics*, vol. 22, no. 13–14, pp. 1539–1558, 2008, https://doi.org/10.1163/156855308X360640.

[16] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, planning, and control for quadruped locomotion over challenging terrain," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 236–258, Feb. 2011, https://doi.org/10.1177/0278364910388677.

[17] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: A review," *Neural Networks*, vol. 21, no. 4, pp. 642–653, May 2008, https://doi.org/10.1016/j.neunet.2008.03.014.

[18] C. P. Santos and V. Matos, "CPG modulation for navigation and omnidirectional quadruped locomotion," *Robotics and Autonomous Systems*, vol. 60, no. 6, pp. 912–927, Jun. 2012, https://doi.org/10.1016/j.robot.2012.01.004.

[19] H. Kimura, Y. Fukuoka, and A. H. Cohen, "Adaptive Dynamic Walking of a Quadruped Robot on Natural Ground Based on Biological Concepts," *The International Journal of Robotics Research*, vol. 26, no. 5, pp. 475–490, May 2007, https://doi.org/10.1177/0278364907078089.

[20] C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, "Motion Planning for Quadrupedal Locomotion: Coupled Planning, Terrain Mapping, and Whole-Body Control," *IEEE Transactions on Robotics*, vol. 36, no. 6, pp. 1635–1648, Dec. 2020, https://doi.org/10.1109/TRO.2020.3003464.

[21] C. D. Bellicoso, F. Jenelten, C. Gehring, and M. Hutter, "Dynamic Locomotion Through Online Nonlinear Motion Optimization for Quadrupedal Robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2261–2268, Jul. 2018, https://doi.org/10.1109/LRA.2018.2794620.

[22] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "ANYmal parkour: Learning agile navigation for quadrupedal robots," *Science Robotics*, vol. 9, no. 88, Mar. 2024, Art. no. eadi7566, https://doi.org/10.1126/scirobotics.adi7566.

[23] P. Arena, L. Patanè, and S. Taffara, "A Data-Driven Model Predictive Control for Quadruped Robot Steering on Slippery Surfaces," *Robotics*, vol. 12, no. 3, Jun. 2023, Art. no. 67, https://doi.org/10.3390/robotics12030067.

[24] P.-A. Léziart, T. Corbères, T. Flayols, S. Tonneau, N. Mansard, and P. Souères, "Improved Control Scheme for the Solo Quadruped and Experimental Comparison of Model Predictive Controllers," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9945–9952, Oct. 2022, https://doi.org/10.1109/LRA.2022.3192581.

[25] M. H. Rahman, S. B. Alam, T. D. Mou, M. F. Uddin, and M. Hasan, "A Dynamic Approach to Low-Cost Design, Development, and Computational Simulation of a 12DoF Quadruped Robot," *Robotics*, vol. 12, no. 1, Feb. 2023, Art. no. 28, https://doi.org/10.3390/robotics12010028.

[26] L. T. Thanh, T. B. Thang, L. V. Cuong, and H. T. T. Binh, "Multitask Augmented Random Search in deep reinforcement learning," *Applied Soft Computing*, vol. 160, no. C, Jul. 2024, Art. no. 111605, https://doi.org/10.1016/j.asoc.2024.111605.

[27] R. Kurazume, S. Hirose, and K. Yoneda, "Feedforward and feedback dynamic trot gait control for a quadruped walking vehicle," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation*, Seoul, South Korea, May 2001, pp. 3172–3180 vol.3, https://doi.org/10.1109/ROBOT.2001.933105.

[28] V.-T. Nguyen *et al.*, "Robust adaptive nonlinear PID controller using radial basis function neural network for ballbots with external force," *Engineering Science and Technology, an International Journal*, vol. 61, Jan. 2025, Art. no. 101914, https://doi.org/10.1016/j.jestch.2024.101914.

[29] V.-T. Nguyen, D.-N. Duong, D.-H. Phan, T.-L. Bui, X. HoangVan, and P. X. Tan, "Adaptive Nonlinear PD Controller of Two-Wheeled Self-Balancing Robot with External Force," *Computers, Materials & Continua*, vol. 81, no. 2, pp. 2337–2356, Nov. 2024, https://doi.org/10.32604/cmc.2024.055412.