IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

FT-CPG: Learning Central Pattern Generators for Fault-Tolerant Quadruped Locomotion under Multi-Joint Failures

Pei Zhang[1], Zhaobo Hua[1], Qiyu Qiu[1] and Jinliang Ding[1]

***Abstract*—Quadruped robots used for rescue and exploration are susceptible to various leg failures, where unpredictable joint locking or power loss can pose an immediate risk of falling. Traditional controllers lack fault-tolerant control capabilities in the case of multi-joint concurrent faults, and erroneous controller outputs may lead to robot damage. This paper proposes a model-free reinforcement learning framework based on central pattern generators (CPG) for fault-tolerant control (FT-CPG). The framework uses biomimetic gait generation and section-wise training to address various types of multi-joint concurrent faults. FT-CPG adopts a fault-tolerant CPG module to generate safe gaits, while utilizing neural network-based policies to infer failures and coordinate the rhythmic behaviors of the CPG, ensuring the ability to track velocity commands under fault conditions. Experiments show that FT-CPG is robust in unexpected situations, where a single leg experiences failures across any number of joints, with each joint randomly encountering locking or power loss faults. Furthermore, the proposed framework preserves the robot's omnidirectional mobility. Finally, zero-shot sim-to-real transfer was successfully implemented on the real-world Unitree Go1 robot, effectively addressing various multi-joint leg failures.**

***Index Terms*—Legged robots, Reinforcement learning, AI-based methods.**

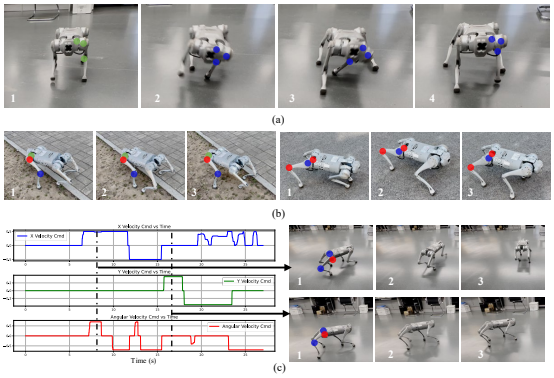

Fig. 1. Fault-tolerant control performance of FT-CPG on the Unitree Go1 robot. Specifically, (a) illustrates a robot in a healthy state during locomotion that suddenly loses balance due to power-loss faults affecting all three joints of the front left leg. The proposed framework enables rapid recovery of locomotion. (b) demonstrates the robot's ability to maintain mobility outdoors despite encountering different types of joint failures in the front left and hind right legs. (c) showcases the robot successfully tracking omnidirectional velocity commands even after experiencing a combination of joint faults in the right hind leg. In these figures, red markers indicate joint locking failures, blue markers represent power-loss failures, and green markers denote healthy joints.

I. INTRODUCTION

QUADRUPED robots are increasingly utilized in rescue and exploration missions owing to their remarkable mobility and stability. However, in dangerous environments, these robots are susceptible to encountering unforeseen leg joint failures, such as mechanical damage, motor power loss, or joint locking. These failures can severely compromise the robot's mobility, posing both safety risks and economic losses. Equipping robots with effective fault-tolerant control strategies to autonomously handle unforeseen leg joint failures can significantly enhance safety and mitigate the impact of these failures. Common joint failures include joint locking and power loss. Joint locking can cause a robot's joint to suddenly lock at a fixed angle during operation, preventing it from being adjusted by control signals. Joint power loss refers to the inability of the motor to provide sufficient torque. Both failures induce substantial changes in the robot's dynamic model, necessitating the development of sophisticated control strategies to maintain balance and mobility under such conditions [1], [2].

Current research on fault-tolerant control for joint failures mainly includes traditional methods and learning-based methods. Traditional methods focus on dynamic modeling and gait planning. These existing methods require fault detection algorithms [3], complex mathematical modeling [4], and gait planning for specific faults [5], enabling the robot to adapt to altered dynamics following failures. Learning-based approaches use reinforcement learning (RL) algorithms to train control policies within simulators. These approaches have proven effective in addressing simple joint locking [6] or power loss failures [7], [8]. However, much of the existing literature concentrates on idealized scenarios involving single joint failures or specific fault types. In practice, robots often face situations where multiple joints experience different types of faults simultaneously, making fault-tolerant control more challenging. Furthermore, most of the current research predominantly focuses on enabling the robot to move forward

Manuscript received: January, 17, 2025; Revised March, 18, 2025; Accepted May, 8, 2025.

This paper was recommended for publication by Editor Jens Kober upon evaluation of the Associate Editor and Reviewers' comments. This work was supported in part by the National Key R&D Plan Project under Grant 2022YFB3304700 and the Xingliao Talent Program of Liaoning Province of China under Grants XLYC2202002. (Corresponding author: Jinliang Ding)

[1]Pei Zhang, Zhaobo Hua, Qiyu Qiu and Jinliang Ding are with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China pei.zhang088@outlook.com; Bobby_1752635630@outlook.com; qiu_qiyu_neu@163.com; jlding@mail.neu.edu.cn;

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

after failures, with less emphasis on maintaining omnidirectional mobility.

In contrast, quadrupedal animals in nature exhibit remarkable robust fault-tolerant control abilities. When faced with disabilities, these animals are able to restore their original mobility by relying on their remaining limbs. This fault-tolerant control capability is closely related to the regulation of movement by the nervous system in animals, encompassing both central pattern generators (CPGs) within the spinal cord and reflex circuits originating from the cerebral cortex [9], [10]. In recent years, CPGs have been widely applied to gait control in quadruped robots, demonstrating significant potential for adaptive control [11]–[14].

In this paper, we propose a model-free reinforcement learning fault-tolerant control framework based on central pattern generators (CPG), called FT-CPG, to provide quadruped robots with fault-tolerant control capabilities akin to those of animals. The framework consists of four novel components: (1) a fault-tolerant CPG module that generates gait signals, (2) a healthy RL policy responsible for producing nominal gaits under healthy conditions, (3) a fault RL policy that adjusts the gaits in the presence of faults and generates residual joint positions to emulate reflex circuits for rapid responses to sudden failures, and (4) a leg state discriminator that evaluates the condition of multiple joints to determine the appropriate gait. To evaluate the effectiveness of the proposed method, we conduct extensive simulations and hardware experiments, as illustrated in Fig. 1, we demonstrate that FT-CPG enables the Unitree Go1 [15] to rapidly regain locomotion following unexpected failures. The framework effectively handles multi-joint concurrent faults within a single leg and maintain omnidirectional mobility, rather than merely surviving or following fixed-speed commands. The main contributions of this paper are summarized as follows:

- The FT-CPG framework is proposed as a fault-tolerant control solution, enabling robots to actively perceive and handle joint failures. This work extends previous fault-tolerant control research by addressing various multi-joint concurrent faults within a single leg.
- A CPG module for fault-tolerant control is designed, which can be adjusted by two independent RL policies. Safe gaits are generated under various fault conditions through section-wise training, without relying on complex constraints.
- Omnidirectional motion command tracking is achieved under failure conditions. Even in fault states, the robot is enabled by the controller to track body linear velocity along the x and y directions, as well as angular velocity around the z axis.
- Comprehensive sim-to-real experiments are carried out with the Unitree Go1, showcasing the successful implementation of zero-shot sim-to-real transfer for fault-tolerant control using the FT-CPG algorithm.

II. RELATED WORK

A. Fault-Tolerant Control for Joint Failures

1) **Traditional Methods**: Traditional fault-tolerant control methods for joint failures typically rely on model-based approaches and system identification techniques [2], [3], [16]. These methods first utilize system identification to determine the fault type, followed by building a complex mathematic model to design the fault-tolerant gait [17], [18]. However, the identification and modeling process can be time-consuming and labor-intensive, making it challenging to address unknown faults. Furthermore, the gait design process is generally tailored to specific types of robots or failures. For example, Du et al. [16] propose a fault-tolerant gait for the six-legged robot, using screw theory and Jacobian matrices to enable movement with broken legs. Similarly, full-body control algorithms and geometric model optimization have been applied to design fault-tolerant gaits for quadruped robots [2], [19]. These model-based manual gait design methods lack generalization to different types of failures, limiting its applicability.

2) **Learning-Based Methods**: Existing learning-based fault-tolerant control approaches primarily focus on addressing single joint faults [6]–[8], [20]. For example, a student-teacher network architecture is proposed [6] to handle single joint lock failures in quadruped robots, but it is limited to tracking a fixed speed. In [8], a random joint masking method is introduced, incorporating gait rewards and curriculum learning to maintain locomotion and turning capabilities after a single joint experiences locking or weakening failures. Similarly, [7] introduces a fault-tolerant control network, utilizing supporting polygon heuristic rewards and joint limit constraints (e.g., setting the minimum angle of the calf joint to 2.1 rad to prevent excessive inward leg swinging) to address motor power loss in robots. However, the joint constraints require extensive experimentation and manual tuning. Hou et al. [21] propose a switching control scheme for dual-joint, dual-fault scenarios in a single leg, where an appropriate pre-trained controller is selected based on the fault type. While this approach provides an initial solution for multi-joint failures, it assumes that each joint experiences the same type of fault, which is often unrealistic. In contrast, our approach is designed to handle multiple, randomly occurring fault types across several joints while preserving the robot's omnidirectional mobility.

In this paper, we extend our focus from single-joint or single-type fault scenarios to cases involving multiple joints and types of faults within a single leg. Different combinations of concurrent failures significantly increase the difficulty of fault-tolerant control. Our method enables the quadruped robot to overcome these failures, generate reliable gait patterns, and maintain its original omnidirectional velocity tracking capability.

B. Biomimetic CPG Control Methods

Inspired by neuroscience, research on biomimetic controllers focuses on simulating the locomotion patterns of legged vertebrates to enhance the motion capabilities of robots. Among these approaches, central pattern generators (CPGs) have attracted significant attention as a promising method. CPGs are neural circuits in the vertebrate nervous system that can spontaneously generate rhythmic motor patterns, such as walking or crawling, through internal oscillations. These neural circuits can also be modulated by higher-level neural

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

systems, enabling organisms to produce more complex and adaptive behaviors [9].

Some studies have successfully combined reinforcement learning with CPG oscillators to develop effective locomotion controllers for robots [11]–[14], [22]–[24]. For example, [14] employs a CPG-based teacher-student learning framework to enable quadrupedal robots to adapt to uneven terrains. In [13], a simple policy-modulated trajectory generator is introduced to facilitate rapid gait learning and velocity control in quadrupedal robots. Similarly, [24] constructs a muscle excitation model using CPGs and combines it with deep reinforcement learning to develop a control policy for soft robots. Compared to direct RL-based independent joint position control, the CPG-based approach takes into account the interrelationships among joints, facilitating faster learning and improved coordination. Nevertheless, in dynamic environments, CPGs typically need to be integrated with reflex control mechanisms to achieve greater adaptability and flexibility [10]. Previous studies [10], [25] have successfully combined CPGs with reflex control to facilitate the adaptation of humanoid and quadrupedal robots to rough terrains in simulation.

In this work, drawing inspiration from CPGs and reflex mechanisms, we design a CPG module specifically for fault-tolerant control and combine it with RL algorithms to address multi-joint and multi-type concurrent failures in quadruped robots. Compared to existing methods, our approach offers additional advantages in fault conditions: Building upon the healthy gait generated by the healthy policy, the fault policy fine-tunes the gait under failure conditions and superimposes subtle residual joint offsets. This mechanism prevents excessive movements without the need for manually predefined special joint constraints. Moreover, unlike prior approaches that rely on curriculum learning or complex gait reward functions, our method simplifies the training process while maintaining robust fault tolerance.

III. METHOD

A. Overview

The dynamics of a robot exhibit significant differences between healthy and fault states, making it difficult to learn a single, universal policy. As illustrated in Fig. 2, our FT-CPG framework adopts a section-wise [26] training approach to learn two independent policies, the healthy policy $\pi^h(\boldsymbol{a}^h|\boldsymbol{s}^h)$ and the fault policy $\pi^f(\boldsymbol{a}^f|\boldsymbol{s}^f)$, which jointly control a fault-tolerant CPG module. The module receives amplitude and frequency signals from both policies, generates foot trajectories, and outputs desired joint positions through inverse kinematics (IK). Subsequently, PD controllers are used to output motor torques, enabling motion control under both healthy and various fault conditions.

In the first phase, the healthy policy π^h is trained to enable the CPG module to generate periodic amplitude and phase signals, facilitating fundamental gait generation and omnidirectional velocity control under healthy conditions. Building upon this stable gait, the second phase involves training the fault policy π^f, which is designed to handle various failure scenarios. This policy re-coordinates the amplitude and phase signals of the CPG while providing residual joint position offsets $\Delta\boldsymbol{q}$, allowing the robot to generate fault-tolerant gaits with rapid response capabilities. Simultaneously, a discriminator is trained to predict leg states based on the robot's historical states, assisting the policies in performing fault-tolerant control.

B. Fault Tolerant CPG Module

In this work, we propose a fault-tolerant CPG module based on the oscillator model presented in the literature [12]. Our module consists of four leg amplitude-phase oscillators, which are modulated by a reinforcement learning policy and can generate diverse stable gaits under different fault conditions:

$$\ddot{r}_{x_i} = a_x(\frac{a_x}{4}(g(\mu_{x_i}) - r_{x_i}) - \dot{r}_{x_i}), \tag{1}$$

$$\ddot{r}_{y_i} = a_y(\frac{a_y}{4}(g(\mu_{y_i}) - r_{y_i}) - \dot{r}_{y_i}), \tag{2}$$

$$\dot{\theta}_i = g(\omega_i) + \frac{1}{2}\sum_j (r_{x_j} + r_{y_j}) w_{ij} \sin(\theta_j - \theta_i - \phi_{ij}), \tag{3}$$

where i represents the leg index ($i = 1, 2, 3, 4$), r_{x_i} and r_{y_i} are the current amplitudes of the oscillator, and θ_i is the phase of the oscillator. a is a positive convergence factor, couplings between different oscillators are defined by the weights $w_{i,j}$ and phase biases $\phi_{i,j}$.

Compared to previous work, we extend the CPG to fault-tolerant control scenarios by introducing the fault-tolerant intrinsic amplitude and frequency, denoted as $g(\mu_i)$ and $g(\omega_i)$, respectively. The calculation process for these parameters is as follows:

$$g(\mu_{x_i}) = \text{clip}(\mu^h_{x_i} + \mu^f_{x_i}, \mu_{\min}, \mu_{\max}), \tag{4}$$

$$g(\mu_{y_i}) = \text{clip}(\mu^h_{y_i} + \mu^f_{y_i}, \mu_{\min}, \mu_{\max}), \tag{5}$$

$$g(\omega_i) = \text{clip}(\omega^h_i + \omega^f_i, \omega_{\min}, \omega_{\max}). \tag{6}$$

The fault-tolerant intrinsic amplitude and frequency consist of both healthy and fault signals, where the healthy signals $\boldsymbol{\mu}^h$ and $\boldsymbol{\omega}^h$ are provided by the healthy policy $\pi^h(\boldsymbol{a}^h|\boldsymbol{s}^h)$, and the fault signals $\boldsymbol{\mu}^f$ and $\boldsymbol{\omega}^f$ are provided by the fault policy $\pi^f(\boldsymbol{a}^f|\boldsymbol{s}^f)$. The healthy and fault signals are summed together and then clipped to a predefined range. This combined signal effectively drives the CPG module to generate diverse gaits, allowing the robot to adapt to both healthy and various joint fault conditions. We set $\mu_{\min} = 1$, $\mu_{\max} = 2$, $\omega_{min} = 0$ Hz, $\omega_{\max} = 120$Hz, $a_x = a_y = 100$.

As shown in Fig. 3, the states of oscillators are mapped to the local coordinate system of each leg. The transformation process from CPG to the local coordinate system is as follows:

$$x_{i,d} = -d(2\frac{r_{x_i} - \mu_{\min}}{\mu_{\max} - \mu_{\min}} - 1)\cos(\theta_i), \tag{7}$$

$$y_{i,d} = l_1 + d(2\frac{r_{y_i} - \mu_{\min}}{\mu_{\max} - \mu_{\min}} - 1)\cos(\theta_i), \tag{8}$$

$$z_{i,d} = \begin{cases} -h + g_c \sin(\theta_i), \text{if } \sin(\theta_i) > 0, \\ -h + g_p \sin(\theta_i), \text{otherwise}, \end{cases} \tag{9}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

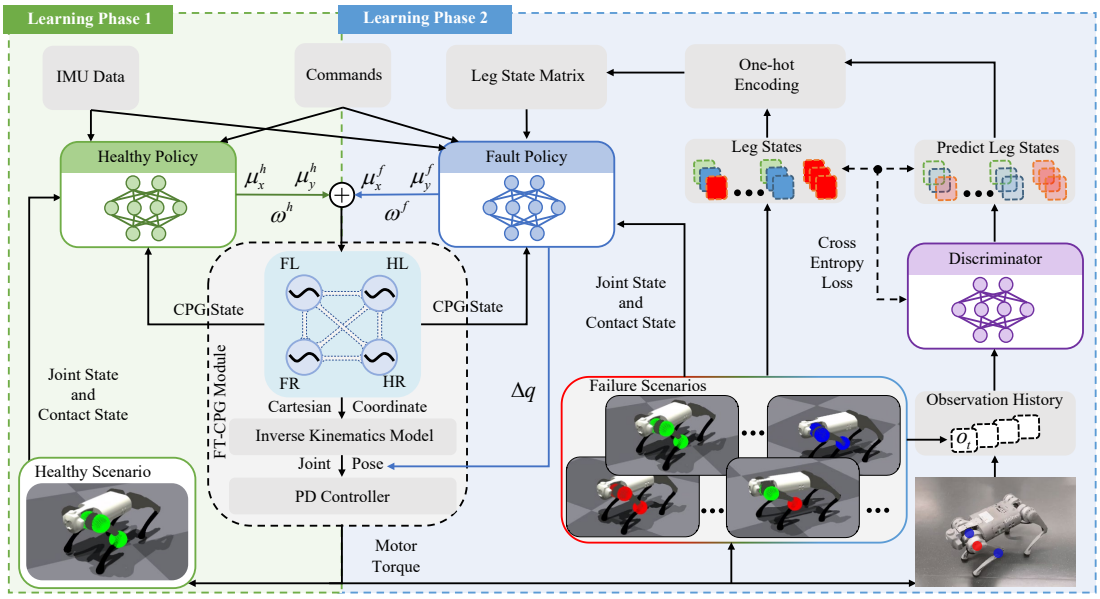

Fig. 2. FT-CPG Fault-Tolerant Control Framework. Our control framework consists of two neural network control policies, a shared fault-tolerant CPG module, and a fault discriminator. Two policies provide amplitude and frequency, with the CPG module generating footstep trajectories, the healthy policy provides intrinsic frequency and amplitude signals to generate nominal foot-end trajectories under normal conditions. Building upon this, the fault policy adjusts the intrinsic frequency and amplitude while introducing additional residual joint position offsets to rapidly generate fault-tolerant gaits. Simultaneously, the discriminator network predicts the fault type based on historical observations, supplying the fault policy with a leg state prediction matrix. The CPG module contains four oscillators, corresponding to four legs (front left (FL, $i = 1$), front right (FR, $i = 2$), hind left (HL, $i = 3$), hind right (HR, $i = 4$)).

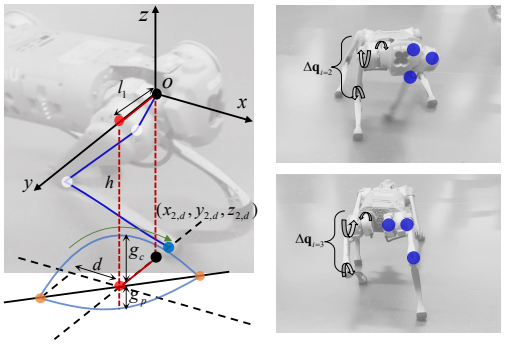

Fig. 3. The left-side image illustrates the process of mapping the CPG states to the Cartesian coordinate system. In contrast, the right-side image depicts the application of residual joint position signals to one of the robot's legs under fault conditions.

where d is the maximum step length, h is the body height, g_c and g_p represent the swing and support heights, respectively. l_1 is the length of the robot's hip. Subsequently, the target joint positions of the three motors of the i-th leg can be obtained using inverse kinematics:

$$\boldsymbol{q}_i^{target} = IK(x_{i,d}, y_{i,d}, z_{i,d}) + \Delta \boldsymbol{q}_i, \tag{10}$$

where $\Delta \boldsymbol{q}_i \in [-0.5, 0.5]$ rad are the residual joint positions of i-th leg provided by $\pi^f(\boldsymbol{a}^f|\boldsymbol{s}^f)$.

C. Training Architecture

We define the training process for the fault-tolerant policies as a Markov Decision Process (MDP), represented by the tuple $M = \{\mathcal{S}, \mathcal{A}, R, \mathcal{P}\}$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ is the action space, and $\mathcal{P}$ represents the state transition function. The policies are defined as the mapping from states to actions, i.e., $\pi(\boldsymbol{a}|\boldsymbol{s})$. The agent observes the environmental states and samples actions through policies, then interacts with the environment, and receives rewards R. By maximizing the total reward per episode using the Proximal Policy Optimization (PPO) algorithm [27], we aim to learn the optimal policies π^{h*} and π^{f*}. The training is carried out in two phases:

1) **Learning Phase 1**: As depicted in Fig. 2, we train a healthy policy $\pi^h(\boldsymbol{a}^h|\boldsymbol{s}^h)$ in healthy environments. This policy enables the robot to track body linear velocities in the x direction ranging from $[-0.5, 0.5]$ m/s, in the y direction from $[-0.4, 0.4]$ m/s, and angular velocity in the z-axis from $[-0.8, 0.8]$ rad/s. In this phase, only gait patterns under healthy conditions are learned, without considering fault scenarios. Therefore, the computation of the intrinsic amplitude and frequency within the CPG module relies solely on the output of $\pi^h(\boldsymbol{a}^h|\boldsymbol{s}^h)$, and the target joint angle calculation does not require residual position offsets. Specifically, the terms $\mu_{x_i}^f, \mu_{y_i}^f, \omega_i^f$ and $\Delta \boldsymbol{q}_i$ in equations (4), (5), (6), and (10) are set to zero.

2) **Learning Phase 2**: To enable the robot to adapt to multi-joint failure scenarios and track the same velocity commands, we keep the parameters of the healthy policy π^{h*} fixed and focus on learning a universal fault policy $\pi^f(\boldsymbol{a}^f|\boldsymbol{s}^f)$. This fault policy is trained across scenarios encompassing both healthy and faulty conditions. In these scenarios, each joint in the robot's leg is assigned three possible states, corresponding to three labels: health (label 0), locking fault (label 1), and power loss fault (label 2). The locking and power loss faults are simulated based on the failure mechanisms described in references [6] and [7].

We sample the state labels for each joint with a probability

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

of 1:1:1 at the beginning of each episode. Given that there are three joints in a single leg, 27 possible state combinations can be formed, and each combination is referred to as a "leg state" (e.g., $\boldsymbol{z}_l = (1, 2, 0)$). These leg states are then one-hot encoded into a leg state matrix $\boldsymbol{Z} \in \mathbb{R}^{3\times3}$ (e.g., $\boldsymbol{z}_l \rightarrow \boldsymbol{Z} = [0, 1, 0; 0, 0, 1; 1, 0, 0]$). It should be noted that we primarily focus on fault-tolerant control for multiple joint failures within a single leg. As a result, the algorithm requires training separate fault policies for different fault-affected legs.

To accelerate the learning process, we employ parallel training across 4096 environments, thereby enabling online learning in the presence of multiple fault conditions. Additionally, to enable the robot to actively detect failures and generate fault-tolerant gaits, we also train a discriminator capable of identifying different types of failures. This discriminator processes short-term historical proprioception data as input and outputs $\hat{\boldsymbol{Z}} \in \mathbb{R}^{3\times3}$, representing the probability of each of the three joints being in a healthy, locked, or power loss state. During deployment, we determine the most probable predicted leg state $\hat{\boldsymbol{z}}_l$ based on $\hat{\boldsymbol{Z}}$ and compute a mask ($mask \leftarrow \hat{\boldsymbol{z}}_l == [0, 0, 0].$) This mask selectively blocks or allows the output of π_f, formulated as: $\boldsymbol{a}_f \leftarrow (1-mask)\cdot\boldsymbol{a}_f$. This masking mechanism effectively exploits the advantages of the section-wise training process, ensuring a clear separation between gaits in healthy and faulty states.

D. Configuration of Training and Environment

Observation Space: We define two independent observation spaces, $\mathcal{S}^h$ and $\mathcal{S}^f$, corresponding to the two training stages.

For the first stage, the observation $\boldsymbol{s}^h \in \mathbb{R}^{73}$ includes the IMU data (base angular velocity, projected gravity), command velocities, joint positions and velocities, foot contact boolean, last action of π^h, and CPG states $\{\boldsymbol{r}_x, \dot{\boldsymbol{r}}_x, \boldsymbol{r}_y, \dot{\boldsymbol{r}}_y, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\}$.

In the second stage, the observation $\boldsymbol{s}^f \in \mathbb{R}^{106}$ extends $\boldsymbol{s}^h$ by incorporating the last action of the fault policy π^f and the leg state matrix $\boldsymbol{Z}$. During this stage, the input of the discriminator consists of the historical state of $\boldsymbol{s}^h$ with a length of 10.

Action Space: We define two separate action spaces, $\mathcal{A}^h$ and $\mathcal{A}^f$, for the two training stages.

For the first stage, the action generated by $\pi^h(\boldsymbol{a}^h|\boldsymbol{s}^h)$ consists of the intrinsic amplitude and frequency of the CPG module, represented as $\boldsymbol{a}^h = [\boldsymbol{\mu}_x^h, \boldsymbol{\mu}_y^h, \boldsymbol{\omega}^h] \in \mathbb{R}^{12}$. This action primarily controls the frequency and phase difference between the legs, forming a basic omnidirectional gait pattern in the healthy state.

For the second-stage training, the fault policy $\pi^f(\boldsymbol{a}^f|\boldsymbol{s}^f)$ refines the basic gait by adjusting the intrinsic amplitude and frequency to establish a new rhythmic pattern under fault conditions. Simultaneously, it directly outputs the residual joint angles for all 12 joints, simulating the biological reflex circuit to enhance the robot's robustness against external disturbances. Thus, the action $\boldsymbol{a}^f$ comprises the intrinsic amplitude and frequency of the CPG, and residual joint positions, formulated as: $\boldsymbol{a}^f = [\boldsymbol{\mu}_x^f, \boldsymbol{\mu}_y^f, \boldsymbol{\omega}^f, \Delta\boldsymbol{q}] \in \mathbb{R}^{24}$.

Reward Function: Our reward function primarily rewards the tracking of the commands of body linear velocities in the x and y directions, as well as the angular velocity. We utilized the same reward function used in the previous studies [12]. Notably, there is no need to design specialized fault-tolerant rewards or curriculum learning mechanisms.

Training Details: The training process of FT-CPG is conducted in the NVIDIA Isaac Gym simulator [28], using the Unitree Go1 quadruped robot model. We perform parallel training with 4096 robots, randomly placed on undulating terrain within the range of -4 to 4 cm. Both the healthy and fault policy networks are implemented as Multi-Layer Perceptrons (MLPs) with layer sizes of [512, 256, 128]. The discriminator is based on a single-layer Transformer Encoder [29]. The control frequency of policies is set to 50Hz, while the torques are updated at 200Hz, and the equations governing each oscillator are also integrated at a rate of 200Hz. The gains of the PD controller for the joints are set to K_p=30.0 and K_d=0.6. The training is conducted on an NVIDIA RTX3090 desktop GPU. Each episode lasts for 20 seconds, with the first phase of training taking approximately 20 minutes, and the second phase of training requiring around 1.5 hours.

IV. EXPERIMENT RESULTS

A. Fault locomotion in Isaac Gym Simulation

We evaluate our proposed method with the baseline RL PD control method [30], the RL-based fault-tolerant control method 'Random joint masking' [8], and variants of the proposed method. For the evaluation, we set body velocity commands as follows: from 0-5s, track a velocity command of $\pm$ 0.5 m/s in the body x-direction; from 5-10s, track a velocity command of $\pm$ 0.4 m/s in the body y-direction; from 10-15s, track an angular velocity command of $\pm$ 0.6 rad/s. The survival rate and omnidirectional velocity tracking results for all methods are recorded, as shown in Tables I, II, and Fig. 4. 'No-Residual joint position' means that the residual joint position $\Delta\boldsymbol{q}$ to the CPG module is removed. 'No-Fault CPG input' indicates that the CPG signals $(\boldsymbol{\mu}_x^f, \boldsymbol{\mu}_y^f, \boldsymbol{\omega}^f)$ from the fault policy are removed. 'Mixed policy' refers to training one fault policy directly in environments encompassing all leg conditions, without employing a section-wise training process.

TABLE I
SURVIVE RATE UNDER DIFFERENT FAULT CONDITIONS IN ISSAC GYM SIMULATION

Fault leg	Number of Faulty joints	Baseline [30]	Random joint masking [8]	No-Residual joint position	No-Fault CPG input	Mixed policy	Ours
Health	0	100.00	100.00	100.00	100.00	100.00	100.00
Front Left	1	66.67	100.00	43.33	69.33	100.00	100.00
	2	31.08	98.92	27.42	45.17	100.00	100.00
	3	24.25	97.63	12.38	24.63	100.00	100.00
	Avg	40.67	98.85	27.71	46.38	100.00	**100.00**
Hind Right	1	63.50	100.00	40.67	48.33	100.00	100.00
	2	55.17	99.75	25.58	36.92	99.88	100.00
	3	39.88	96.75	16.00	18.83	99.88	100.00
	Avg	52.85	98.83	27.42	34.38	99.88	**100.00**
Total avg		52.67	98.97	35.61	47.00	99.95	**100.00**

1) Survival Rate Evaluation: We set up 2700 parallel robot environments for all 27 fault leg states, with 100 environments for each state, and test the control effectiveness when faults

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

occurred in different legs (FL leg and HR leg). The success rates for all methods are shown in Table I. All methods achieve a 100% survival rate under the healthy condition. However, as the number of faulty joints increases, the survival rate for baseline [30] drops significantly. 'Random joint masking' [8] achieves a 100% survival rate in single-joint fault cases, but the survival rate decreases in more fault scenarios. In contrast, our proposed method maintains a 100% survival rate under all fault conditions. The table also demonstrates the effectiveness of each component in our proposed framework: the residual joint position and fault CPG outputs are crucial for improving survival rates, because the absence of either component prevents the formation of fault-tolerant gaits. In comparison, the lack of section-wise training has a smaller impact on survival rates, with the survival rate for multiple faults being second only to our proposed method.

TABLE II
TRACKING ERROR COMPARISON IN ISSAC GYM SIMULATION

Fault leg	Number of Faulty joints	Metrics	RMSE ↓					
			Baseline [30]	Random joint masking [8]	No-Residual joint position	No-Fault CPG input	Mixed policy	Ours
Health	0	v_x	0.1874	0.1887	0.2399	0.2359	0.2124	0.1880
		v_y	0.1049	0.1213	0.1544	0.1479	0.2581	0.1154
		ω_z	0.2236	0.2430	0.2243	0.2776	0.4050	0.2085
	Avg		0.1720	0.1843	0.2062	0.2205	0.2918	**0.1706**
Front Left	1	v_x	0.3506	0.2152	0.4325	0.2261	0.2474	0.1804
		v_y	0.2477	0.1453	0.2720	0.1695	0.2725	0.1358
		ω_z	0.4637	0.2731	0.5884	0.5048	0.4181	0.1971
	Avg		0.3540	0.2112	0.4310	0.3001	0.3127	**0.1711**
	2	v_x	0.4453	0.2467	0.4577	0.2602	0.2520	0.1923
		v_y	0.2665	0.1481	0.2823	0.2329	0.2769	0.1496
		ω_z	0.5587	0.2898	0.6014	0.5885	0.3861	0.2005
	Avg		0.4235	0.2282	0.4471	0.3605	0.3050	**0.1808**
	3	v_x	0.5134	0.3191	0.4875	0.3373	0.2761	0.2047
		v_y	0.3012	0.1946	0.2940	0.2662	0.2800	0.1646
		ω_z	0.6000	0.4185	0.5961	0.5981	0.4233	0.2025
	Avg		0.4715	0.3107	0.4592	0.4005	0.3265	**0.1906**
Hind Right	1	v_x	0.3477	0.2407	0.4475	0.2594	0.2296	0.1690
		v_y	0.2475	0.1259	0.2817	0.2366	0.2821	0.1416
		ω_z	0.5987	0.2611	0.8130	0.5421	0.3642	0.1862
	Avg		0.3980	0.2092	0.5141	0.3460	0.2920	**0.1656**
	2	v_x	0.4187	0.2446	0.4740	0.3415	0.2207	0.1823
		v_y	0.2736	0.1288	0.2917	0.2733	0.2738	0.1411
		ω_z	0.6627	0.2651	0.6478	0.5464	0.4001	0.1878
	Avg		0.4517	0.2128	0.4712	0.3871	0.2982	**0.1704**
	3	v_x	0.4589	0.2548	0.4757	0.3567	0.2254	0.1937
		v_y	0.2803	0.1503	0.2956	0.2670	0.2692	0.1542
		ω_z	0.5836	0.3068	0.5900	0.5269	0.4511	0.1864
	Avg		0.4409	0.2373	0.4538	0.3835	0.3152	**0.1781**
Total avg			0.3893	0.2311	0.4261	0.3426	0.3059	**0.1758**

2) Omnidirectional Velocity Tracking Evaluation: Fault-tolerant control must not only ensure survival in the case of faults, but also guarantee the accuracy of velocity tracking. We use the Root-Mean-Square Error (RMSE) metric to evaluate the tracking accuracy of each method, with experimental results shown in Table II. The 'Random joint masking' achieves suboptimal performance in the case of a single fault, but the tracking accuracy significantly drops in the case of multiple faults. In contrast, our method achieves the best tracking results under all fault conditions. This table also indicates the necessity of section-wise training: the 'Mixed policy', which lacks section-wise training, can only achieve relatively high survival rates, but its omnidirectional velocity tracking performance is poor. On the other hand, methods lacking residuals and Fault CPG components exhibit similar low tracking accuracy as baseline. Fig. 4 visualizes the tracking of x linear velocity commands during 0-5s for 0 to 3 fault scenarios in the case of FL leg faults. Under the healthy condition, all methods achieve high tracking accuracy, though the 'Mixed Policy' performs noticeably worse. As faults increase, our approach maintains stable tracking, whereas the Baseline and 'No-Residual joint position' lead to robot failure. Meanwhile, 'Random joint masking' retains some tracking capability but exhibits significant velocity oscillations with two or three faults.

The above results indicate that multi-joint concurrent faults across different combinations of a single leg pose significant challenges to previous methods. In contrast, the proposed approach allows the robot to maintain high survival rates and accurate omnidirectional velocity tracking.

B. Results of sudden fault recovery in locomotion capability

In this section, we further analyze how the FT-CPG actively restores and maintains locomotion capability when joint faults occur. The robot is commanded to move at a constant speed of 0.5 m/s, with multiple joint failures deliberately triggered during operation. Fig. 5 illustrates the changes in the joint positions of one healthy leg (FR leg) before and after a concurrent power loss failure occurs in three joints of the FL leg.

Before the fault occurs, the discriminator predicts the leg state as $\hat{\boldsymbol{z}}_l = (0, 0, 0)$ (healthy), and thus the FT-CPG controls the robot's movement using the π^h. All three joints follow the periodic positions generated by the CPG module (green line) as the target positions (red line). When faults occur, the robot loses balance, as the failed FL leg can no longer support the body, resulting in a significant change in the robot's state. The discriminator autonomously detects this anomaly and automatically activates the π^f to rapidly restore the robot's locomotion capability. As shown in the figure, under the influence of $[\boldsymbol{\mu}_x^f, \boldsymbol{\mu}_y^f, \boldsymbol{\omega}^f]$, the periodic trajectory of the position curve originally generated by the CPG is elongated, and its amplitude increases. Meanwhile, the joint residual positions $\Delta \boldsymbol{q}$ exhibit regular oscillations, and the combination of these two factors forms a new target position curve. With this adjustment, the robot regains its balance and continues walking using the remaining three healthy legs. Our accompanying video demonstrates this process and also shows the recovery of the hind legs when damaged.

C. Various Gaits in different fault conditions

The results of Section IV.A show that the section-wise training improves the accuracy of omnidirectional velocity tracking in various fault scenarios. In this section, we further analyze the advantages of the FT-CPG's section-wise training method in gait generation. Fig. 6 visualizes the gaits generated by our proposed method and the Mixed policy without using section-wise training, during the tracking of a 0.5 m/s forward speed in the presence of various faults in the FL and HR legs.

In the healthy state, FT-CPG generates a smooth walking gait for the robot. When the FL leg state is faulty $\boldsymbol{z}_l = (2, 2, 1)$,

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

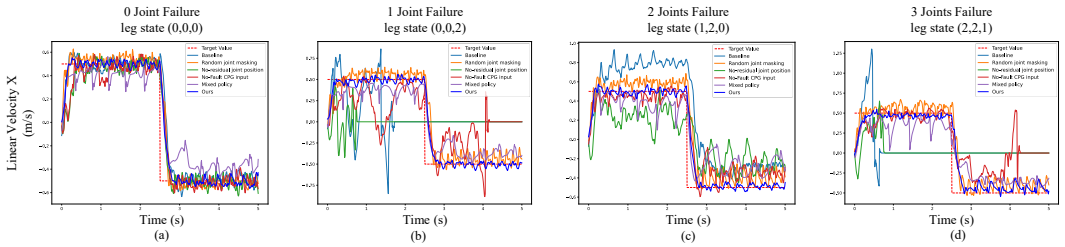

Fig. 4. Velocity commands response curves within 0–5s in the Isaac Gym simulation. We compare the baseline [30], 'Random joint masking' [8], and variants of our proposed method (with different components ablated). Figures (a)–(d) depict velocity tracking performance under 0, 1, 2, and 3 concurrent faults, respectively. A sudden drop to zero in the velocity curve indicates a robot fall.

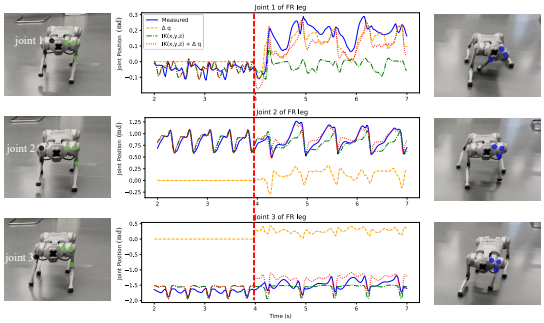

Fig. 5. The joint positions of the FR leg before and after triggering a concurrent power loss failure in three joints of the LF leg.

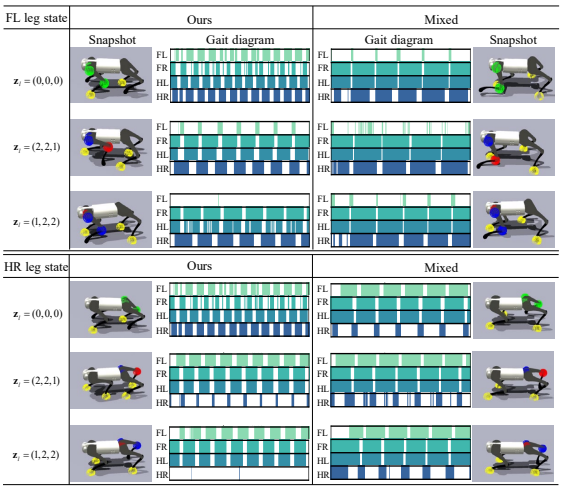

Fig. 6. Time history of the gaits under both healthy and LF and HR legs joint concurrent faults are shown for our method and the one generated by the absence of the section-wise mixed policy. The yellow spheres in the figure represent the moments when the ground contact force at the foot exceeds the 10N threshold.

the robot mainly uses the remaining three healthy legs for walking, with the FL leg occasionally vertically supporting the ground, generating a limping gait. When the FL leg state is $\boldsymbol{z}_l = (1, 2, 2)$, the robot avoids the FL leg from contacting the ground and forms a three-legged gait. In contrast, the Mixed policy mainly uses the FR, HL, and HR legs for walking, both in healthy and faulty states, leading to abnormal gait in the healthy state. Moreover, the gaits formed under the three leg states exhibit severe homogeneity. Fig. 6 also shows the gaits after HR leg failure, and the Mixed policy also exhibits

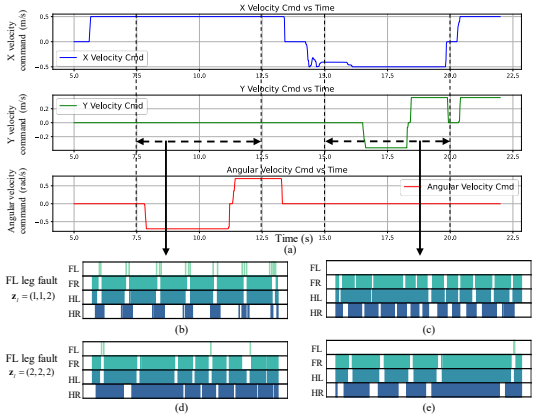

Fig. 7. The proposed method is deployed on the real Go1 robot, generating gait time histories under two different concurrent faults in the FL leg during omnidirectional motion tracking. (a) shows the velocity tracking commands from the joystick input. (b) and (c) illustrate the gaits under the faulty leg state (1,1,2) while tracking forward and turning commands, as well as backward and sideward commands, respectively. (d) and (e) display the gaits under a different faulty leg state (2,2,2).

homogeneity under different failure conditions. The reason for this result is that a single policy struggles to learn different gaits for distant distributions.

In contrast, FT-CPG's section-wise training mode isolates healthy and faulty gaits into two separate policies, avoiding confusion between these gaits. Additionally, with the discriminator actively predicting the leg state as input to the fault policy, FT-CPG generates various gait patterns under different fault conditions.

D. Omnidirectional fault-tolerant gaits on Go1 robot

Fig. 7 shows the hardware experiment results where the Go1 robot, under omnidirectional motion commands, generates different fault-tolerant gaits in the presence of various faults (FL leg for example). From 7.5s to 12.5s, the velocity command instructs the robot to move forward while turning to the right and left. In the leg state $\boldsymbol{z}_l = (1, 1, 2)$, the robot generates the gait pattern shown in Fig. 7b, while in the leg state $\boldsymbol{z}_l = (2, 2, 2)$, the support time of the HR leg significantly increases, generating a new gait (Fig. 7d). From 15s to 20s, the velocity command instructs the robot to move backward while shifting to the right and left. In both fault scenarios, the robot generates different gait patterns.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

These results show that, under the same fault conditions, FT-CPG can generate appropriate fault-tolerant gaits according to different velocity commands. Additionally, under the same velocity command, different fault combinations lead to different fault-tolerant gait patterns. Moreover, we also tested the fault-tolerant control effect of the algorithm on different outdoor terrains, and the specific details can be found in the supplementary video.

V. CONCLUSIONS

In this study, we investigate common leg faults encountered by quadruped robots, considering multiple joints simultaneously experiencing different types of locking or power loss faults, thus extending previous fault-tolerant control works. The proposed FT-CPG framework integrates biomimetic control methods with section-wise reinforcement learning to address various multi-joint concurrent faults. Additionally, our control scheme preserves the robot's omnidirectional movement capability under fault conditions and implements zero-shot sim-to-real transfer on a real Unitree Go1 robot, enabling it to effectively handle a wide range of leg faults.

However, a limitation of our approach is that separate fault policies must be trained for different legs experiencing multi-joint failures. Future work will explore a unified policy to handle multi-leg, multi-joint failures. Other promising extensions for this work in the future include: 1) Integrating the existing fault-tolerant control scheme into various complex terrains to enhance the overall mobility of quadruped robots. 2) Considering fault-tolerant control with sensor failures [31], such as failures in the robot's vision or posture sensors, to improve the generality of fault-tolerant control. 3) Combining offline learning and other methods to design more flexible and reliable fault-tolerant control schemes based on animal movement data [32].

REFERENCES

[1] J.-M. Yang, "Fault-tolerant gaits of quadruped robots for locked joint failures," *IEEE Trans. Syst., Man, Cybern. C*, vol. 32, no. 4, pp. 507–516, 2002.

[2] M. M. Gor, P. M. Pathak, A. K. Samantaray, J. M. Yang, and S. W. Kwak, "Fault-tolerant control of a compliant legged quadruped robot for free swinging failure," *Proc. Inst. Mech. Engineers, Part I: J.syst. Control Eng.*, vol. 232, no. 2, pp. 161–177, 2018.

[3] C. Ferrell, "Failure recognition and fault tolerance of an autonomous robot," *Adapt. Behav.*, vol. 2, no. 4, pp. 375–398, 1994.

[4] J.-M. Yang, "Kinematic constraints on fault-tolerant gaits for a locked joint failure," *J. Intell. Robot. Syst.*, vol. 45, pp. 323–342, 2006.

[5] X. Chen, F. Gao, C. Qi, and X. Tian, "Gait planning for a quadruped robot with one faulty actuator," *Chin. J. Mech. Eng-En.*, vol. 28, no. 1, pp. 11–19, 2015.

[6] D. Liu, T. Zhang, J. Yin, and S. See, "Saving the limping: Fault-tolerant quadruped locomotion via reinforcement learning," 2022, *arXiv:2210.00474*.

[7] Z. Luo, E. Xiao, and P. Lu, "Ft-net: Learning failure recovery and fault-tolerant locomotion for quadruped robots," *IEEE Robot. Autom. Lett.*, vol. 8, no. 12, pp. 8414–8421, 2023.

[8] M. Kim, U. Shin, and J.-Y. Kim, "Learning quadrupedal locomotion with impaired joints using random joint masking," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Yokohama, Japan, May. 2024, pp. 9751–9757.

[9] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: a review," *Neural Netw.*, vol. 21, no. 4, pp. 642–653, 2008.

[10] G. Li, A. Ijspeert, and M. Hayashibe, "Ai-cpg: Adaptive imitated central pattern generators for bipedal locomotion learned through reinforced reflex neural networks," *IEEE Robot. Autom. Lett.*, 2024.

[11] G. Bellegarda and A. Ijspeert, "Cpg-rl: Learning central pattern generators for quadruped locomotion," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 12 547–12 554, 2022.

[12] G. Bellegarda, M. Shafiee, and A. Ijspeert, "Visual cpg-rl: Learning central pattern generators for visually-guided quadruped locomotion," in *in Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Yokohama, Japan, May. 2024, pp. 1420–1427.

[13] A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke, "Policies modulating trajectory generators," in *Proc. Conf. Robot Learn. (CoRL)*, Zürich, Switzerland, Oct. 2018, pp. 916–926.

[14] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Sci. Robot*, vol. 7, no. 62, p. eabk2822, 2022.

[15] U. Robotics, "Go1," Nov. 2021. [Online]. Available: https://www.unitree.com/products/go1/

[16] H. Du and F. Gao, "Fault tolerance properties and motion planning of a six-legged robot with multiple faults," *Robotica*, vol. 35, no. 6, pp. 1397–1414, 2017.

[17] J.-M. Yang, "Fault-tolerant gait planning for a hexapod robot walking over rough terrain," *J. Intell. Robot. Syst.*, vol. 54, pp. 613–627, 2009.

[18] Y. Liu, X. Fan, L. Ding, J. Wang, T. Liu, and H. Gao, "Fault-tolerant tripod gait planning and verification of a hexapod robot," *Appl. Sci.*, vol. 10, no. 8, p. 2959, 2020.

[19] J. Cui, Z. Li, J. Qiu, and T. Li, "Fault-tolerant motion planning and generation of quadruped robots synthesised by posture optimization and whole body control," *Complex Intell. Syst.*, vol. 8, no. 4, pp. 2991–3003, 2022.

[20] C. Chen, C. Li, H. Lu, Y. Wang, and R. Xiong, "Meta reinforcement learning of locomotion policy for quadruped robots with motor stuck," *IEEE Trans. Autom. Sci. Eng.*, pp. 1–15, 2024.

[21] T. Hou, J. Tu, X. Gao, Z. Dong, P. Zhai, and L. Zhang, "Multi-task learning of active fault-tolerant controller for leg failures in quadruped robots," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. Yokohama, Japan: IEEE, May. 2024, pp. 9758–9764.

[22] L. Campanaro, S. Gangapurwala, D. De Martini, W. Merkt, and I. Havoutis, "Cpg-actor: Reinforcement learning for central pattern generators," in *Proc. Annu. Conf. Towards Auton. Robot. Syst. (TAROS)*, Lincoln, UK, September. 2021, pp. 25–35.

[23] G. Li, A. Ijspeert, and M. Hayashibe, "Ai-cpg: Adaptive imitated central pattern generators for bipedal locomotion learned through reinforced reflex neural networks," *IEEE Robot. Autom. Lett.*, vol. 9, no. 6, pp. 5190–5197, 2024.

[24] S. Min, J. Won, S. Lee, J. Park, and J. Lee, "Softcon: Simulation and control of soft-bodied animals with biomimetic actuators," *ACM Trans. Graph. (TOG)*, vol. 38, no. 6, pp. 1–12, 2019.

[25] G. Sun, M. Shafiee, P. Li, G. Bellegarda, A. Ijspeert, and G. Sartoretti, "Learning-based hierarchical control: Emulating the central nervous system for bio-inspired legged robot locomotion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Sys. (IROS)*, Abu Dhabi, UAE, Oct. 2024, pp. 13 938–13 945.

[26] A. I. Károly, P. Galambos, J. Kuti, and I. J. Rudas, "Deep learning in robotics: Survey on model structures and training strategies," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 266–279, 2021.

[27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[28] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu based physics simulation for robot learning," in *Adv. Neural Inf. Process. Syst. (NIPS) Datasets and Benchmarks*, Virtual-only, Dec. 2021.

[29] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, and Y. Wang, "Transformer in transformer," in *Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 34, Virtual-only, Dec. 2021, pp. 15 908–15 919.

[30] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proc. Conf. Robot Learn. (CoRL)*, Auckland, New Zealand, Dec. 2022, pp. 91–100.

[31] C. Zhang, J. Jin, J. Frey, N. Rudin, M. Mattamala, C. Cadena, and M. Hutter, "Resilient legged local navigation: Learning to traverse with compromised perception end-to-end," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Yokohama, Japan, May. 2024, pp. 34–41.

[32] L. Han, Q. Zhu, J. Sheng, C. Zhang, T. Li, Y. Zhang, H. Zhang, Y. Liu, C. Zhou, R. Zhao, *et al.*, "Lifelike agility and play in quadrupedal robots using reinforcement learning and generative pre-trained models," *Nat. Mach. Intell.*, vol. 6, no. 7, pp. 787–798, 2024.