2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Full-Scale Autonomous Highway Inspection with Quadruped Robot: Multi-Level Locomotion Learning in Complex Environments

Chenxiang Ma, Chengcheng Xu, Feng Wang

***Abstract*— This paper proposes an innovative approach of full-scale autonomous highway inspection in complex environments using quadruped robot to enhance the adaptability and coverage of inspection tasks. Considering adaptive locomotion control as the foundation of autonomous inspection, a multi-level locomotion learning framework based on reinforcement learning is developed, including primitive-level, skill-level and inspection-level. Primitive-level control policy built upon Vector Quantized Variational Autoencoder is trained through imitation learning from existing open-source robots locomotion models, thereby achieving discrete embedding and reusability of foundational locomotion knowledge. At skill-level, to support diverse inspection skills learning, parametric modular scenario modeling method of the highway environment is proposed. Each skill-level control network is trained in corresponding modular scenario while reusing primitive-level control network. Inspection-level control network is established through multi-skill distillation from trained skill control networks. Combined with coverage path generator, automatic inspection can be completed. In a simulated complex highway environment, inspection robot demonstrates diverse inspection skills, successfully completing inspection of 14,400m^2 area in 0.4h, with speed of 2.37m/s. Coverage and hazard detection rates both reach 100%. Compared to the existing highway inspection forms, the proposed highway inspection framework with quadruped robot enables efficient, stable, and full-scale autonomous inspection in complex highway environments, which provides general deployment capability for intelligent inspection systems.**

I. Introduction

Regular inspection and maintenance of highways is one of the key tasks to ensure the safe operation and service capabilities. However, due to the complexity of the highway environment, the inspection work faces numerous challenges. In addition to the pavement for vehicles, there are also terrain structures of varying complexity, which impose more restrictions on inspection tasks. Recently, autonomous inspection technology has been gradually applied to highway inspection. The main forms used include road inspection vehicles and unmanned aerial vehicle (UAV) [1]. Although these forms have achieved improved inspection performance and reduced labor costs to a certain extent, their application still have significant limitations. Since inspection vehicles rely on ground driving, their movements are hindered when facing unpaved roads and complex terrain areas such as steep slopes

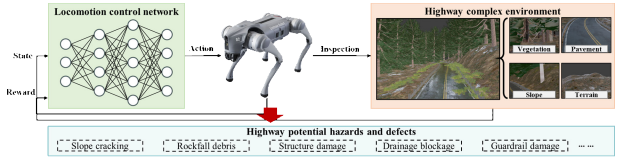


Fig. 1 Highway inspection process of quadruped robot

[2]. Meanwhile, although UAV inspection has greater mobility and can cover a wider range of scenarios, they are affected by endurance, weather conditions, and may encounter navigation difficulties in closed spaces like tunnels. Stability cannot be guaranteed in long-term and long-distance inspection tasks [3]. Therefore, relying solely on existing autonomous inspection technologies still cannot meet the full-scale intelligent inspection requirements in diverse and complex highway environments.

The application of quadruped robot in highway inspection has broad prospects. The highway inspection process based on a quadruped robot is shown in Fig. 1. Through sensing information, locomotion controller will output the motor torque of each joint, and achieve stable autonomous inspection in rough terrain, narrow space and irregular road surface to detect highway hazards and defects. Compared with inspection vehicle or UAV, quadruped robot has stronger terrain adaptability. It is able to flexibly shuttle through various complex highway scenarios, achieving the largest coverage. Meanwhile, even without external sensor information, it can still rely on its own proprioception, reducing the possibility of inspection interruption. At present, quadruped robots have been widely used in many inspection fields such as underground pipeline inspection [4], forest monitoring [5], etc., showing excellent environmental adaptability and continuous working ability. However, in the field of highway inspection, few studies focus on the application of quadruped robot, and inspection behavior learning for complex highway environments needs to be realized [6]. In order to achieve autonomous inspection of complex highway environments based on quadruped robot, there are still the following issues that need to be considered.

First, it is necessary to address the complex and time-consuming development process of locomotion control for highway inspection tasks. Due to the empirical adjustments and simplifications of traditional kinematic and dynamic modeling, simulation-based reinforcement learning (RL) methods have gradually become the mainstream trend of locomotion control [7], where the robot continuously adjusts its action policy through interaction with the environment to maximize cumulative long-term reward. Since direct training may produce unnatural behaviors, existing studies consider imitation learning based on animal motion data to achieve this process, where the robot learns strategies by imitating expert

*Resrach supported by the National Natural Science Foundation of China (No. 52525204 and 52232012), Science and Technology Innovation Program of Xiongan New Area (No. 2023XAGG0089), Science and Technology Research Program of Hebei Province (No. 252F0802D). Corresponding author: Chengcheng Xu.

The authors are with School of Transportation, Southeast University, Si Pai Lou #2, Nanjing, 210096, China (E-mail: machenxiang@seu.edu.cn; xuchengcheng@seu.edu.cn; 220243607@seu.edu.cn).

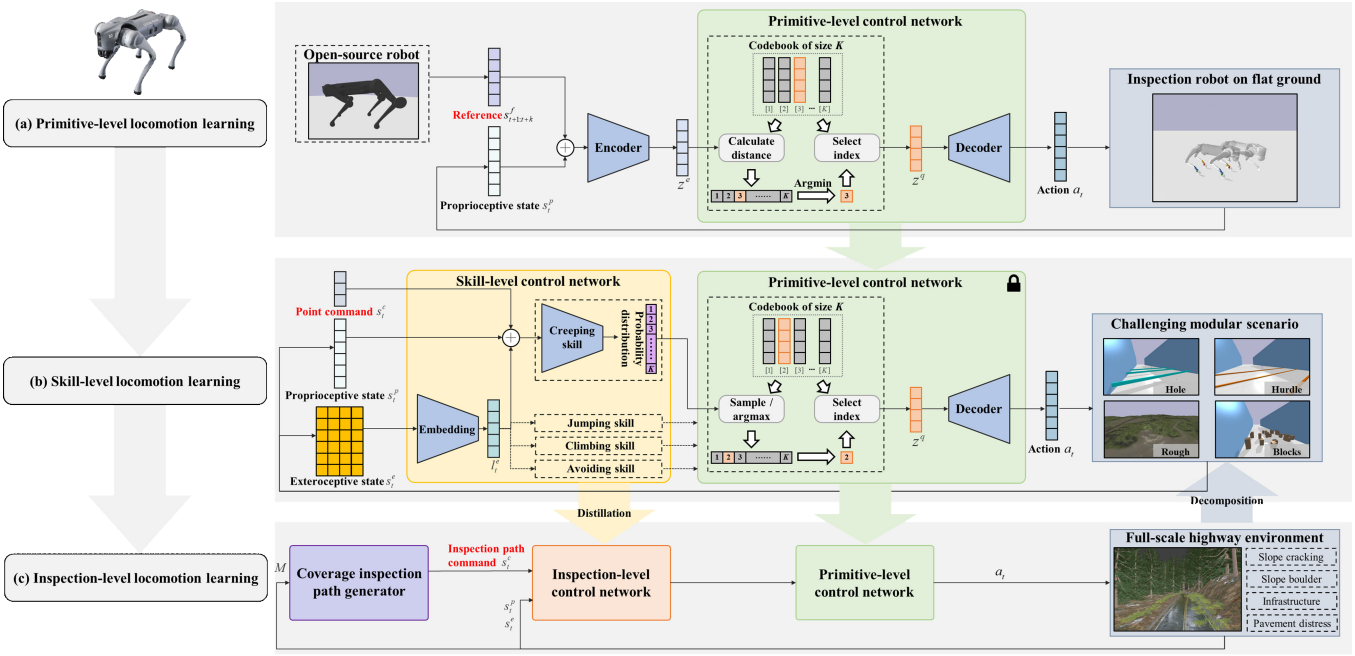


Fig. 2 Highway inspection locomotion learning framework of quadruped robot

demonstration behaviors as reward [8]. Although it exhibits lifelike behaviors, this process takes a lot of time to collect animal motions. In fact, there are already multiple types of robots that have demonstrated behaviors close to animals [9]. It is feasible to skip data collection and directly imitating from trained robots with less time cost.

Secondly, considering inspection scenario modeling, it is necessary to build a high-fidelity training environment to solve the simulation-reality gap problem. In addition to these open and structured environments with high degrees of freedom that previous research mainly focused on [10], overall highway environment also includes complex and changeable high slopes, auxiliary facilities such as drainage ditches. The rough terrain, isolated rocks and vegetation will also hinder normal inspection, which is a scenario that needs to be emphasized during the highway inspection process [11]. In general, the existing inspection scenario modeling is still relatively simple and difficult to achieve transfer to the real world.

Thirdly, facing multiple complex highway scenarios, robots need to possess diverse and generalizable inspection skills. Existing studies are mostly designed for specific individual tasks, resulting in narrow applications. Inspired by biological control systems, complex tasks can be first divided into essential sub-skills that are learned individually and later composed, enabling a progressive enhancement of overall capability [12]. This can also be applied to inspection task with knowledge distillation. Each specific model serves as a teacher, and knowledge is transferred to a single student model by imitating the output distributions of multiple teachers, thereby achieving efficient and unified collaborative control.

In order to achieve autonomous highway inspection with higher adaptability, coverage and safety, this paper focuses on the innovative application of quadruped robot in highway inspection, thereby proposing multi-level inspection locomotion learning framework. At primitive-level, based on the imitation learning of existing open-source robots' trajectory, robot learns basic gait behaviors, forming primitive-level network. At skill-level, complex highway environment is decomposed into challenging modules based on components, enabling parametric representation. Primitive network is reused and extended with multiple skill-level control networks for specific inspection skills. At inspection-level, all skill control networks are distilled into a unified inspection-level control network through multi-skill distillation. An inspection coverage path generator is incorporated to provide command for full-scale inspection. Proposed framework ensures broader coverage and higher adaptability for highway inspection, offering a novel quadruped-based solution beyond existing inspection forms.

The main contributions of this study are as follows.

(1) Establishing a multi-level inspection locomotion learning framework, whose hierarchical representation provides a stable control interface for higher-level tasks, enabling direct invocation without training from scratch.

(2) Proposing imitation learning of primitive behaviors within models, bypassing complex and scarce motion data collection and directly learning parametric knowledge from numerous open-source models.

(3) Achieving specialized multiple inspection skills learning and unified compression. With parametric modular scenario modeling, learned skills can be freely generalized, composed, and reused, mitigating the low training efficiency and catastrophic forgetting of unified training.

II. Methods

Overall framework is shown in Fig. 2. Each stage focuses on different learning tasks and environmental complexity levels, and all are trained through reinforcement learning.

TABLE I Definition of the primitive-level reward function

Component	Formula	Details
Root position reward	$r_t^p = \exp[-20\left\|p_{root}^*(t) - p_{root}(t)\right\|^2 - 10\theta^2], \theta = \text{angle}(R^* R^{-1})$	$p_{root}^*(t)$, R^* : target root position and orientation, $p_{root}(t)$, R : current root position and orientation
Root velocity reward	$r_t^v = \exp(-2\left\|v_{t,root}^* - v_{t,root}\right\|^2 - 0.2\left\|\omega_{t,root}^* - \omega_{t,root}\right\|^2)$	$v_{t,root}^*$, $\omega_{t,root}^*$: target root linear/angular velocity, $v_{t,root}^*$, $\omega_{t,root}^*$: current root linear/angular velocity
End-effector tracking reward	$r_t^e = \exp(-40\sum_e \left\|x_{t,e}^* - x_{t,e}\right\|^2)$	$x_{t,e}^*$: relative target position of the *k*-th toe to the root, $x_{t,e}$: relative current position of the *k*-th toe to the root
Joint position reward	$r_t^{jp} = \exp(-5\sum_j \left\|q_{t,j}^* - q_{t,j}\right\|^2)$	$q_{t,j}^*$: target angle of the *j*-th joint, $q_{t,j}$: actual angle of the *j*-th joint
Joint velocity reward	$r_t^{jv} = \exp(-0.1\sum_j \left\|\dot{q}_{t,j}^* - \dot{q}_{t,j}\right\|^2)$	$\dot{q}_{t,j}^*$: target angular velocity of the *j*-th joint, $\dot{q}_{t,j}$: actual angular velocity of the *j*-th joint

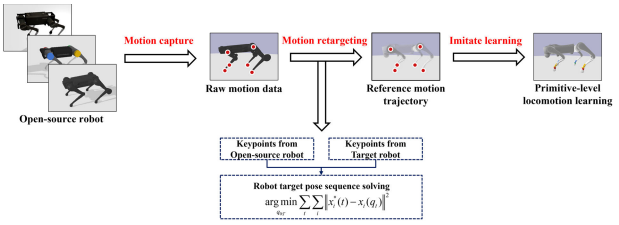


Fig. 3 Expert data collection and processing

A. Primitive Learning Based on Expert Imitation

1) Reference Trajectories from Open-Source Robots: As shown in, we aim for the inspection robot to learn autonomous locomotion abilities from a variety of mature and bio-inspired behaviors. Unlike existing studies that rely on the labor-intensive process of collecting animal motion data from the real world, we directly selected multiple open-source quadruped robots that have achieved flexible locomotion as imitation experts of our inspection robot.

Specifically, as shown in Fig. 3, we selected the A1 and Laikago robots from [8] and the MAX robot from [13]. The simulation environment is constructed using Pybullet [14] as a flat terrain. Each robot will be assigned to perform five types of gaits in the simulation environment: walking, running, backward, spinning, and jumping. The root pose (position and orientation) and the positions (angles) of 12 joints are captured at a speed of 120 fps per minute. Due to morphological differences, collected trajectories must be retargeted. We apply inverse kinematics (IK) to map source motions onto the target robot's kinematic structure, using formulation as follows [15].

$$\arg\min_{q_{0:T}} \sum_t \sum_i \left\| x_i^*(t) - x_i(q_t) \right\|^2 \tag{1}$$

$x_i^*(t)$ represents the 3D position of keypoint i on the open-source robot at each time step t. The source keypoints include positions of feet and hips (8 in total), which are paired with the corresponding target keypoints on the inspection robot. Keypoints of the inspection robot is $x_i(q_t)$, which is determined by pose q_t , including the root orientation and joint position. Then inverse kinematics will be applied to solve pose sequence $q_{0:T}$ of inspection robot.

2) Primitive-Level RL Problem Formulation: Primitive locomotion learning is formulated as a RL process. At each time step t , the robot observes state s_t , takes action a_t according to policy π , receives reward r_t from environment, and moves to the next state s_{t+1} . Above process is repeated until the termination condition is reached, completing an episode and generating a trajectory τ . The objective is to optimize π to maximize expected cumulative reward $J(\pi)$.

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)}[\sum_{t=0} \gamma^t r_t] \tag{2}$$

$$p(\tau \mid \pi) = p(s_0) \prod_{t=0} \pi(a_t \mid s_t) p(s_{t+1} \mid s_t, a_t) \tag{3}$$

where $\gamma \in [0,1]$ represents the discount factor, $p(\tau \mid \pi)$ represents the probability of trajectory τ under given policy π , $p(s_0)$ represents the initial state distribution, and $p(s_{t+1} \mid s_t, a_t)$ represents the state transition probability.

RL components are defined as follows.

Environment: Flat terrain with reference robot.

State: s_t consists of two parts. Proprioceptive state of robot $s_t^p = (q_{t-2:t}, \dot{q}_{t-2:t}, a_{t-2:t})$ contains pose, velocities and historical action in the past three time steps. The pose of a single time step contains root position, root orientation and joint angles. Reference $s_{t+1:t+k}^f$ contains reference root positions, orientations and joint positions for next k time step.

Action: The action a_t is the residual target position of each joint. The current action is added to the current joint position to form the target positions, which are then passed to Proportional-Derivative (PD) controller [16] that ultimately outputs motor torque. The control policy queries new action at 100 Hz, while control frequency of PD controller is 500 Hz.

Reward function: The reward at the primitive-level will encourage policy to track and imitate the reference motion trajectory at each time step, which is defined as follows, with specific details provided in the Table I.

$$r_t = w^p r_t^p + w^v r_t^v + w^e r_t^e + w^{jp} r_t^{jp} + w^{jv} r_t^{jv} \tag{4}$$

TABLE II Definition of the skill-level reward function

Component	Formula	Details
Direction following reward	$r_t^{dir} = \exp[-5(1-\cos(\theta_{yaw}-\theta_{target}))^2]$	θ_{yaw} : current yaw angle of robot's root, $\theta_{target} = \arctan 2(d_y^*, d_x^*)$: angle corresponding to target
Distance progress reward	$r_t^{dis} = \exp[-10(\frac{d_t - d_{t-1}}{d_0})^2]$	d_0 : initial distance between robot and target, d_{t-1} , d_{t-1} : residual distance of previous step and current step
Speed following reward	$r_t^{vel} = \begin{cases} \exp[-4(\overline{v}-v^*)^2], d_t < \varepsilon \\ 0, \text{otherwise} \end{cases}$	$\overline{v}$: average speed along target direction within the episode, v^* : expected target speed

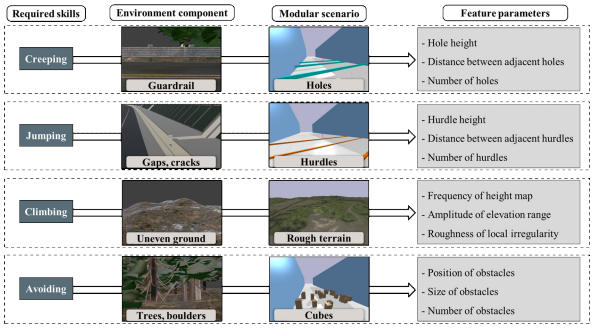


Fig. 4 Construction of challenging modular scenario

Policy: The architecture of control policy is based on Vector Quantized Variational Autoencoder (VQ-VAE), with specific details provided in next section.

Terminations: An episode is terminated if: (1) Robot's roll or pitch exceeds threshold; (2) Full reference trajectory has been executed; (3) Reference and robot diverge extremely.

Optimization algorithm: Proximal Policy Optimization (PPO) algorithm [17] is adopted to optimize the policy.

3) Primitive-Level Locomotion Control Policy Based on VQ-VAE: During primitive-level, we aim for the inspection robot not only to imitate diverse basic gait behaviors, but also to retain this prior knowledge for reuse in subsequent highway inspection tasks. Inspired by generative models, we introduce VQ-VAE [18] as architecture of control policy π . VQ-VAE is a model that combines autoencoder structure with discrete latent space. It compresses and reuses data by encoding high-dimensional features into discrete latent vectors.

The structure is shown in Fig. 2 (a). Different from generative task, VQ-VAE in this work outputs action by inputting the state. Specifically, the encoder takes s_t as input, and outputs latent vector $z^e \in \mathbb{R}^{1\times D}$. A learnable latent embedding space (codebook) $\mathbf{e} \in \mathbb{R}^{K\times D}$ is defined, consisting of D-dimensional vectors $e_1, e_2, e_3 ... e_K$. To achieve vector quantization, distance between z^e and all e_i is calculated, and the closest embedding e_k is selected as the representation, namely z^q . Subsequently, the decoder then takes the latent embedding vector z^q as input and outputs the action a_t .

C. Specific Skill Learning in Challenging Modular Scenarios

1) Challenging Highway Environment Modules: Considering complexity and diversity of highway environments, it is difficult to achieve adaptive and stable locomotion behavior by directly learning locomotion in full-scale scenarios. Therefore, we decompose it into multiple challenging modular scenarios as separated simulation environments for training, enabling abstract representation of components. Parameter randomization is also introduced. Each scenario is associated with a parameter vector $c^T \in C$. During each episode in training, the system randomly samples c^T from space C to dynamically generate scenarios with different feature combinations.

As shown in Fig. 4, design of each scenario is as follows.

(1) Holes, represent gaps underneath infrastructure such as guardrails. Since they are usually too high to step over directly, robot must creep through narrow spaces. Key parameters include hole height, distance between holes, and numbers.

(2) Hurdles, correspond to structures requiring jumping, including gaps and raised features. Key parameters include hurdle height, distance between hurdles, and numbers.

(3) Rough terrain, corresponds to unpaved surfaces such as slopes, requiring robot to maintain balance and achieve stable climbing. Feature parameters include frequency, amplitude, and roughness. Perlin noise [19] is used to generate this scene.

(4) Cubes, simulate potential obstacles encountered during inspection, such as fallen rocks, trees. Obstacles will hinder the robot's inspection path, requiring it to develop avoidance and detour capabilities. Key parameters include number, size, and position of obstacles on a two-dimensional plane.

2) Skill-Level RL Problem Formulation: Four modular scenarios constructed above serve as simulation environments, within which separate skill-level inspection locomotion controllers are trained. Detailed configuration is as follows.

Environment: For each skill control policy, environment is generated by above-mentioned modular scenarios that are specific to each policy, respectively completing the training.

State: s_t consists of three parts. Proprioceptive state s_t^p , identical to primitive-level. Command state s_t^c , which contains target horizontal position and expected speed. Exteroceptive state s_t^e from environment simulates sensing information, including a 2D grid height map, a 2D front depth map and a uniformly distributed rays emitted horizontally across 360°.

Action: It is consistent with the primitive level.

Reward: Reward at skill-level encourages robot to follow the command and reach the target, which is defined as follows, with specific details provided in the Table II.

$$r_t = w^{dir} r_t^{dir} + w^{dis} r_t^{dis} + w^{vel} r_t^{vel} \tag{5}$$

Algorithm 1: Coverage inspection path generation

Input: grid map M, start position (x_0, y_0).
Output: coverage path list P.

$H, W \leftarrow$ shape(M)
$P \leftarrow [\,]$
$(x, y) \leftarrow (x_0, y_0)$
$d \leftarrow 1$
While True **do**
  P.append((x, y))
  $M(x, y) \leftarrow 1$
  $(x', y') \leftarrow (x, y + d)$
  if $0 \le y' < H$ and $M(x', y') == 0$ **then**
    $(x, y) \leftarrow (x', y')$
  else
    $d \leftarrow -d$
    $(x', y') \leftarrow (x + 1,\ y)$
    if $0 \le x' < W$ and $M(x', y') == 0$ **then**
      $(x, y) \leftarrow (x', y')$
    else
      break
return P

Policy: Retrieve the codebook and decoder from frozen primitive-level control policy, and add training-required skill-level control network to form skill-level control policy. Details are provided in the next section.

Terminations: An episode is terminated under any of the following conditions. (1) Instability. This is consistent with primitive-level. (2) Step limit. The maximum number of steps is reached. (3) Target reached. Two-dimensional Euclidean distance between robot and target is less than 0.5m, indicating successful task completion.

3) Skill-*Level Locomotion Control Policy:* After primitive-level control policy is trained, codebook and decoder are reused as primitive-level control network. A skill-level control network is added to output a categorical distribution over discrete latent embeddings to drive primitive-level control network, thereby forming control policy at skill level. The architecture is illustrated in Fig. 2 (b).

Specifically, skill-level control policy is composed of a general environmental embedding network and separate skill-specific sub-network. Exteroceptive state is first input into embedding network, forming a low-dimensional compact representation l_t^e. Subsequently, l_t^e along with command s_t^c and proprioceptive state s_t^p are input into each specific skill sub-network to produce latent embedding distribution p, which represents the selection probabilities over vectors in the codebook. A specific embedding vector e_k is obtained by sampling and input into the frozen primitive-level control network to generate final action.

D. Coverage Inspection Learning via Multi-Skill Distillation

1) Coverage Inspection Path Generator: During highway inspection, the inspection robot navigates through a series of inspection points that collectively cover the entire environment. Specifically, boustrophedon-style coverage strategy [20] is adopted to provide path commands. Target area is represented as a grid map, and robot traverses the environment row by row in alternating directions, thereby ensuring full coverage. Notably, this study does not assume prior knowledge of obstacle locations during path generation. Path generator merely provides an idealized sequence of coverage target points. During actual execution, robot actively generate an appropriate avoidance behavior for obstacles.

As shown in Algorithm 1, environment is modeled as a grid map M. Current point is continuously added to the path P while checking direction. At each step, the robot attempts to move one grid cell to right/left depending on current direction. When next row exceeds boundary, the entire coverage is complete. The final output is a list P of target points to be visited in sequence. Additionally, a random speed command is appended as part of input.

2) Inspection-*Level Locomotion Control Policy:* In skill-level, multiple specific skill-level control networks have been trained for different scenarios. Furthermore, we propose inspection-level control network for full-scale highway inspection tasks, which fuses multiple skill networks into a unified policy through a multi-skill distillation [21].

Specifically, for each skill-level expert network π_{expert}^{i}, complete and successful episodes are executed from corresponding modular scenarios to generate a series of trajectories. The input state s_t and corresponding embedding probability distribution $p_{expert}^{i}(\cdot \mid s_t)$ at each time step t are recorded. All teacher data are aggregated into a teacher dataset.

$$\mathcal{D} = \bigcup_i \mathcal{D}^{(i)} \quad \mathcal{D}^{(i)} = \{(s_t, p_{expert}^{i}(\cdot \mid s_t))\}_{t=1}^{T} \tag{6}$$

The inspection-level control network π_{scene} is trained to output embedding distribution $p_{scene}(\cdot \mid s)$ that approximates the expert distribution given the same input. The Kullback–Leibler (KL) divergence [22] is used as the loss function:

$$\min_{\pi_{scene}} \mathbb{E}_{(s, p_{expert}) \sim \mathcal{D}} [D_{KL}(p_{expert}(\cdot \mid s) \,\|\, p_{scene}(\cdot \mid s))] \tag{7}$$

The trained inspection-level control network will be used alongside the frozen primitive-level control network to generate final action, enabling long-range, stable, and adaptive inspection across complex highway environments.

III. EXPERIMENTS

We adopt Unitree Go2 quadruped robot to implement our autonomous highway inspection framework. In the following simulation experiments, $K_p = 50.0$, $K_d = 0.5$ in PD control. Maximum joint torque is limited to 23.7 Nm. In each training episode, foot-end friction force is randomly sampled from $\mathcal{U}(0.4, 1.0)$. Simulation time step is set to 1/500 s, aligned with the PD control loop.

A. Simulation Environment

Simulation environment for the primitive-level remains plane ground. All other environmental parameters are kept at default values, and no randomization is applied. In skill-level stage, friction coefficient will be sampled from 0.2 to 0.7. Specific parameter spaces and target command for each scenario are determined by robot size. The target point is set 15m-20m from the initial position. The velocity command is randomly selected from 0.5m/s to 3m/s and remains constant throughout a single trail. In inspection-level, a high-fidelity

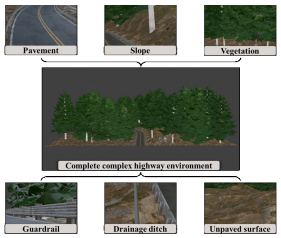


Fig. 5 Inspection experiment scenario

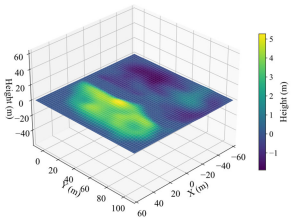


Fig. 6 Terrain height map

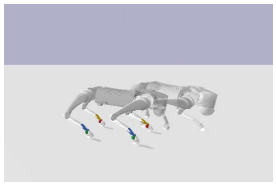


Fig. 7 Snapshot of primitive-level training

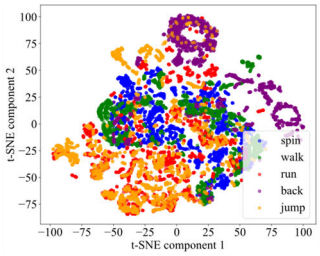


Fig. 8 Visualization of motion trajectories

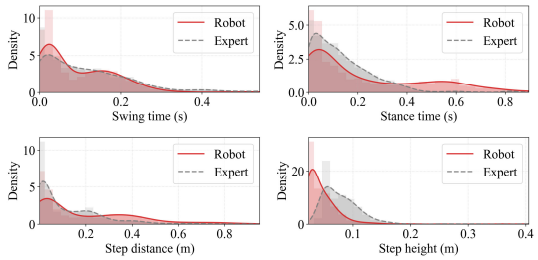


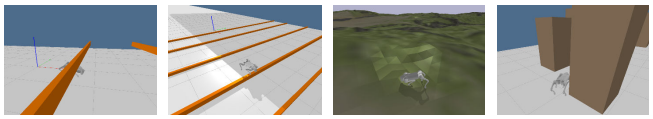
Fig. 9 Comparison of gait parameter distributions

highway simulation environment is constructed, as shown in Fig. 5. The scenario contains typical components, effectively covering various environment modules used during skill-level. Fig. 6 illustrates details of the scenario. Total area spans approximately 120 m × 120 m, with terrain elevation ranging from −1.93m to 5.33m, reflecting prominent topographical undulations including both gentle slopes and abrupt terrain transitions. Friction coefficient of the scenario is set as 0.7. The speed command is set to a constant 2 m/s.

B. Network Implementation

The VQ-VAE used in the primitive-level is designed as follows. The encoder consists of two fully connected layers with hidden dimensions of [256, 256]. The decoder is a three-layer fully connected network with hidden units [128, 256, 256] The value function uses the same architecture as the encoder. Size of codebook is set as 256 with code dimension as 32. Reward weight is set as $w^{p} = 0.15$, $w^{v} = 0.1$, $w^{e} = 0.1$, $w^{jp} = 0.6$, $w^{jv} = 0.05$. Training was conducted on a computational platform equipped with 14-core Intel i9-12900 processor, 32GB of memory, and GeForce RTX 3060 GPU with 12GB of memory. Training process lasted 4 days.

The skill-level control network is constructed as follows. 2D height map and depth map from input state are processed using four layers of 2D convolutional networks, with kernel sizes of 1×1@4 , 4×4@4 , 2×2@4 and 2×2@1, respectively. Outputs are then flattened. 1D radar vector is encoded via a cyclic convolutional structure, consisting of one cyclic convolution layer (4@4) followed by three standard Conv1D



(a) Holes (b) Hurdles (c) Rough terrain (d) Cubes

Fig. 10 Snapshots of skill-level training

TABLE III Performance evaluation of skill-level learning

	Holes	Hurdles	Rough terrain	Cubes
Success rate (%)	74.3	90.2	81.7	83.0
Speed (m/s)	1.790	2.018	1.690	2.363

layers (4@4, 4@4, 4@1). Three types of perception embeddings are concatenated to form the final l_t . Then, s_t^p , s_t^c , l_t are concatenated and passed through a 256-dimensional fully connected layer, a 32 hidden units LSTM layer, and a final fully connected layer that outputs a categorical distribution vector. Reward weight is set as $w^{dir} = 0.2$, $w^{dis} = 0.2$, $w^{vel} = 1.0$. Training platform is consistent with primitive-level. Each scenario takes a total of 3 days.

Inspection-level control network shares the same architecture as skill-level control network. The distillation training process is implemented through supervised learning. Specifically, each pre-trained skill-level control network is used to collect 200 successful full trajectories in corresponding modular scenario, forming teacher dataset. With the same platform, total training time is approximately 4 hours.

IV. Results and Discussions

A. Primitive Behaviors

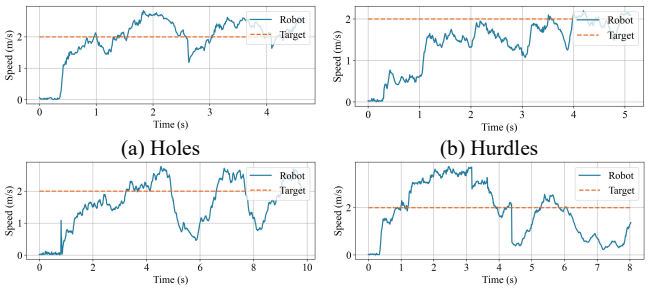
Fig. 7 shows a snapshot of the inspection robot's imitating expert motions during the primitive-level training, indicating a stable learning process. To evaluate the expressive capability of the primitive-level control policy, we use t-SNE technique [23] to visualize the output vectors from the penultimate layer of control policy. The high-dimensional representations are projected onto a two-dimensional space. As shown in Fig. 8, results demonstrate that representative gaits exhibit good separability in embedding space, indicating that robot has learned to form feature boundaries among different behaviors.

We also compare the distribution of key parameters between robot and expert, as shown in Fig. 9, which is described based on all legs. The similar distribution patterns indicate that the robot has successfully captured the expert's dynamics, validating the success of the imitation learning process in primitive-level. However, there are also differences, such as the robot's step height being shorter in certain cases. This could be due to the robot prioritizing higher stability and safety, resulting in more conservative steps.

B. Inspection Skills

Snapshots of the training process under different challenging environment modules are shown in Fig. 10. It can be seen that the inspection robot has demonstrated the inspection skills of creeping, jumping, climbing and avoiding.

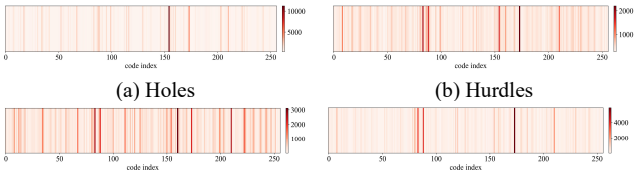
Effectiveness of skill-level learning is evaluated by success rate and average speed over 200 trials, shown in Table III, demonstrating robust task accomplishment and environmental adaptability across various challenging



(a) Holes (b) Hurdles

(c) Rough terrain (d) Cubes

Fig. 11 Speed curves during skill-level execution



(a) Holes (b) Hurdles

(c) Rough terrain (d) Cubes

Fig. 12 Embedding vector selection distributions

scenarios. The success rates remain above 75% and average speed stabilize between 1.7 m/s and 2.3 m/s, indicating that the inspection robot can maintain high task efficiency while ensuring stability.

To verify the inspection robot's compliance with command, Fig. 11 shows speed curves of one trail. It can be seen that speed closely aligns with command. Simultaneously, the frequent fluctuations observed result from the robot sacrificing speed to maintain superior stability and balance when encountering environmental disturbances such as obstacles and holes. This demonstrates that speed is not the primary objective for the robot, but rather the smooth and stable arrival at the target position.

To further analyze performance differences across various skills, we plotted the distribution of embedding vector selections from the codebook across different skill-level networks in Fig. 12, revealing significant distinctions. In holes and cubes scenarios, movements primarily consist of walking with brief periods of creeping or avoiding, resulting in a more concentrated vector selection. In contrast, hurdles and rough terrain scenarios demand continuous adjustments to posture and gait to maintain normal movement. This necessitates the activation of a wider variety of motion vectors, resulting in a more dispersed distribution. Overall, learned skills demonstrate diversity and specialization, ensuring robot possesses a high degree of adaptability and flexibility to handle diverse challenges in highway environments.

C. Full-Scale Highway Inspection Task

Based on the terrain scale and the robot's perception range, the map is divided into 5 m × 5 m grid cells. Safety boundary of 0.25 m is set. Using the proposed path generator, a coverage inspection path was generated with 5 m intervals between targets, starting from the lower-left corner. Resulting path is shown in Fig. 13. The inspection path density reaches 0.18 m/m^2 , demonstrating spatial uniformity and high coverage.

Combined with generated inspection sequence and inspection-level controller, complete inspection task is executed, as illustrated in Fig. 14. 10% of points are randomly

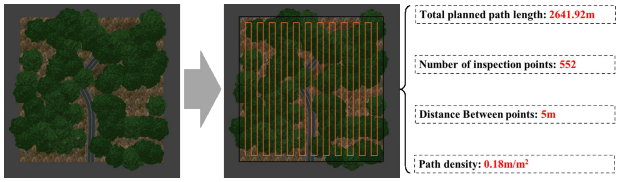


Fig. 13 Planned inspection path information

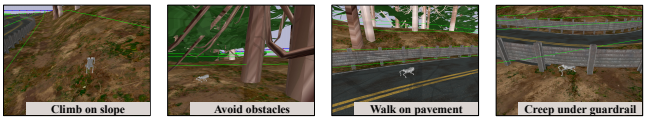
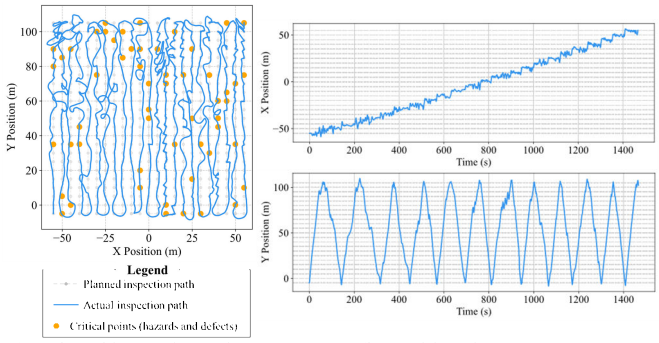


Fig. 14 Snapshots of inspection task



(a) Path and inspection points (b) Position-time curve

Fig. 15 Visualization of actual inspection path

TABLE IV Inspection effect statistical indicators

Robot	Path length (m)	Total time (s)	Average speed (m/s)	Coverage rate (%)	Resets
Go2 (our)	3272.40	1446.85	2.26	100.00	37
A1 [8]	3687.17	2298.42	1.60	86.96	62
Laikago [8]	3864.56	2679.21	1.44	82.79	69
MAX [13]	3598.22	1944.37	1.85	91.12	48

marked as critical points to simulate potential hazard requiring detection. When inspection robot falls or takes more than 1000 steps between two points, task is reset, with next initial position consistent with last position. Skip current target directly when unable to reach it five times in a row.

Overall inspection statistical indicators are provided in Table IV. Inspection robot can successfully complete full coverage of environment, achieving 100% coverage rate, demonstrating that proposed inspection framework can reliably detect potential hazards. Actual path is slightly longer than the planned path, due to additional detours caused by obstacle avoidance or cumulative errors. Average inspection speed is also consistent with performance achieved by different individual skill-level policies. This confirms that unified inspection-level network can effectively inherit locomotion capabilities of each skill-level network, without suffering from performance degradation due to compression.

We simultaneously selected open-source robots imitated in primitive-level to directly perform inspection tasks without undergoing subsequent skill training or compression. Overall, their inspection performance was subpar, exhibiting not only lower efficiency due to longer detours but also lower coverage rates caused by frequent interruptions and failures. This demonstrates that our inspection robot does not merely mimic existing models but achieves superior performance after acquiring various specialized skills, exhibiting unique behaviors adapted for highway inspections.

At microscopic level, robot's 2D spatial path and time-position curve are shown in Fig. 15. The robot's actual execution path aligns with planned inspection path in space and covers all predefined critical points, achieving high-density coverage of entire environment and detecting all hazards. However, compared with theoretical path, the actual path deviates significantly in several areas, with noticeable oscillations, backtracking, or detouring behavior near some inspection points. In addition to encountering unknown obstacles, another reason is that these abnormal fluctuations are concentrated in the inspection failure area. When the robot faces extremely difficult terrain, such as the slope surface at $x = 5$ and $x = 10$, it is difficult to maintain stable movement, causing it to fall off the expected inspection path. Upon resetting, robot must return from failed position to the normal inspection path, thus generating additional abnormal path.

Further combined with time-position curve shown in Fig. 15 (b), X-direction position increases monotonically over time, indicating that robot generally progresses from left to right with stable speed; Y-direction shows regular sawtooth fluctuations, reflecting robot's back-and-forth scanning behavior within each row. This feature is consistent with path planning logic, and also shows that robot responds promptly to commands, maintaining good navigation execution rhythm.

V. Conclusions and Future Work

In order to achieve autonomous inspection in complex highway environments with higher adaptability, coverage and safety, this paper proposes an innovative inspection form of quadruped robot with multi-level inspection locomotion learning framework. In a highway simulation environment with 14400m^2, robot exhibits diverse skills and achieves full-scale task in 0.4h, with coverage rate of 100%. Proposed inspection framework has significant advantages in efficiency, stability and generalization ability over existing methods.

However, due to legal, safety regulations, and testing approval processes, finding suitable highway environments in real world for testing is extremely challenging. Nevertheless, we are making every effort to transition towards empirical experiments. We have identified several feasible sites and are closely collaborating with local authorities to overcome the barriers. Future work will focus on development of sim2real techniques and validation in real-world environments. Excellent performance in simulations has shown great potential of applying quadruped robot in highway inspection, providing a solid foundation for real deployment and holding significant practical application value.

References

[1] S. Yenikaya, G. Yenikaya, and E. Düven, "Keeping the vehicle on the road: A survey on on-road lane detection systems," *ACM Computing Surveys*, vol. 46, no. 1, pp. 1–43, Oct. 2013.

[2] S. Varadharajan, S. Jose, K. Sharma, L. Wander, and C. Mertz, "Vision for road inspection," in *IEEE Winter Conference on Applications of Computer Vision*, Steamboat Springs, CO, USA, Mar. 2014, pp. 115–122.

[3] W. Wu, M. A. Qurishee, J. Owino, I. Fomunung, M. Onyango, and B. Atolagbe, " Coupling deep learning and UAV for infrastructure condition assessment automation," in *2018 IEEE International Smart Cities Conference (ISC2)*, 2018, pp. 1–7.

[4] Y. Jang, W. Seol, K. Lee, K. Kim, and S. Kim, "Development of quadruped robot for inspection of underground pipelines in nuclear power plants," *Electronics Letters*, vol. 58, no. 6, pp. 234–236, Mar. 2022.

[5] M. Mattamala et al., "Autonomous forest inventory with legged robots: system design and field deployment, " *arXiv preprint arXiv:2404.14157*, 2024.

[6] 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.

[7] B. Singh, R. Kumar, and V. P. Singh, "Reinforcement learning in robotic applications: a comprehensive survey," *Artificial Intelligence Review*, vol. 55, no. 2, pp. 945–990, Feb. 2022.

[8] B. Xue, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," in *Robotics: Science and Systems*, Corvalis, Oregon, USA, Jul. 2020.

[9] G. A. Pratt, "Legged robots at MIT: what's new since Raibert?," *IEEE Robotics & Automation Magazine*, vol. 7, no. 3, pp. 15–19, 2000,.

[10] A. Gajanayake, G. Zhang, T. Khan, and H. Mohseni, "Postdisaster impact assessment of road infrastructure: State-of-the-Art review," *Natural Hazards Review*, vol. 21, no. 1, p. 03119002, Feb. 2020.

[11] L. Tan, S. Deng, H. Huang, and R. He, "Research on intelligent inspection of high slopes on Yuqian Road based on UAV," in *International Conference on Smart Transportation and City Engineering (STCE 2022)*, Chongqing, China, Dec. 2022, p. 94.

[12] C. Yang, K. Yuan, Q. Zhu, W. Yu, and Z. Li, "Multi-expert learning of adaptive legged locomotion," *Science Robotics*, vol. 5, no. 49, p. eabb2174, Dec. 2020.

[13] L. Han et al., "Lifelike agility and play in quadrupedal robots using reinforcement learning and generative pre-trained models," *Nature Machine Intelligence*, vol. 6, no. 7, pp. 787–798, Jul. 2024.

[14] A. Iqbal, Y. Gao, and Y. Gu, "Provably stabilizing controllers for quadrupedal robot locomotion on dynamic rigid platforms, " *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 4, pp. 2035–2044, 2020.

[15] M. Gleicher, "Retargetting motion to new characters," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, New York, NY, USA, 1998, pp. 33–42.

[16] P. M. Wensing, M. Posa, Y. Hu, A. Escande, N. Mansard, and A. D. Prete, "Optimization-based control for dynamic legged robots," *IEEE Transactions on Robotics*, vol. 40, pp. 43–63, 2024.

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

[18] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2017, pp. 6309–6318.

[19] A. Lagae et al., "A survey of procedural noise functions," *Computer Graphics Forum*, vol. 29, no. 8, pp. 2579–2600, Dec. 2010.

[20] H. Choset, "Coverage of known spaces: the boustrophedon cellular decomposition," *Autonomous Robots*, vol. 9, no. 3, pp. 247–253, Dec. 2000.

[21] H. Zhang, D. Chen, and C. Wang, "Confidence-aware multi-teacher knowledge distillation," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Singapore, Singapore, 2022, pp. 4498–4502.

[22] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[23] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research,* vol. 9, no. 86, pp. 2579–2605, 2008.