

Article

Skeleton-Based Guidance and Human-in-the-Loop Supervision in VLM Adaptation for Active Vision Defect Inspection

Chenghao Huang¹, Rui Qian¹, Liqiao Xia^{1,2,*} and Pai Zheng^{1,2}

¹ Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hong Kong SAR, China

² Shenzhen Research Institute, The Hong Kong Polytechnic University, Shenzhen 518057, China

* Correspondence: li-qiao.xia@polyu.edu.hk

How To Cite: Huang, C.; Qian, R.; Xia, L.; et al. Skeleton-Based Guidance and Human-in-the-Loop Supervision in VLM Adaptation for Active Vision Defect Inspection. *Human-Machine Interaction* **2026**, *1*(1), 2.

Received: 25 April 2026
Revised: 2 August 2026
Accepted: 17 August 2026
Published: 24 August 2026

Abstract: Defect inspection is critical to industrial safety and reliability, where passive vision with full-area inspection remains limited by partial occlusion and restricted camera fields of view. Currently, active vision provides a promising alternative by enabling adaptive real-time viewpoint adjustment. However, existing active vision methods still suffer from large data requirements and limited generalization across materials, textures, and imaging conditions. Vision-language model (VLM) has shown strong potential for intelligent visual reasoning, motivating the development of a tailored fine-tuning strategy for active vision inspection. Accordingly, this paper proposes an active vision inspection framework through VLM adaptation. First, a skeleton-based guidance method is introduced to estimate the main propagation axis (MPA) of defects and provide a robust geometric prior for viewpoint planning. Subsequently, a human-in-the-loop supervision pipeline is developed to establish high-quality expert inspection priors for model adaptation. Based on these priors, adversarial low-rank adaptation (AdLoRA) is employed to adapt a pre-trained VLM for active vision inspection. Experimental results show that the proposed method outperforms the compared methods in prediction accuracy and improves inspection efficiency relative to passive vision.

Keywords: defect inspection; active vision; vision-language model; adversarial LoRA fine-tuning

1. Introduction

Defect inspection is critical to industrial safety and reliability, and active vision is particularly relevant in industrial scenarios characterized by irregular morphology, occlusion, and limited fields of view [1–4]. In practical systems, passive vision based on full-area inspection usually relies on predefined trajectories or exhaustive viewpoint traversal, which leads to redundant observations and low inspection efficiency. By contrast, active vision selects the next viewpoint from visual feedback and is therefore better suited to efficient inspection in cluttered industrial environments [5–7].

Existing active vision inspection methods can be broadly divided into direction estimation and policy learning. The former infer the main propagation axis (MPA) of the target defect from geometric cues using principal component analysis (PCA) [8] or skeleton-based extraction [9, 10]. However, both methods remain sensitive to segmentation quality, especially under noise, low contrast, and structural interference. The latter predict the next viewpoint directly from observations, often by formulating active vision as a Partially Observable Markov Decision Process (POMDP) and learning policies with deep reinforcement learning (DRL) [5, 7, 11]. Although these approaches can reduce redundant inspection, DRL-based methods require substantial data, as motion policies are learned through interaction rather than direct supervision, and often exhibit poor generalization across materials, textures, and imaging conditions.



Recently, vision-language model (VLM) has demonstrated great potential for active vision inspection, as their multimodal representations enable the integration of visual understanding with action prompts [12–16]. However, the capability of pre-trained VLM is often insufficient for reliable active vision inspection under domain-specific tasks, while full-parameter fine-tuning is costly and less practical for limited task-specific data. Under these conditions, parameter-efficient fine-tuning (PEFT) provides a practical solution for VLM adaptation [17, 18]. But standard PEFT shows limited robustness under domain shift and visual disturbance in active vision inspection, which motivates a more robust adaptation strategy.

To address these challenges, this paper proposes an active vision framework that integrates Skeleton-Based Guidance, Human-in-the-loop Supervision, and robust VLM adaptation, with crack inspection as the target application scenario. Specifically, Skeleton-Based Guidance first provides a geometric prior for crack-direction estimation, which is then verified and refined through Human-in-the-loop Supervision to establish task-specific expert inspection priors. Based on these priors, Adversarial Low-Rank Adaptation (AdLoRA) is employed to improve the robustness of VLM adaptation for active vision inspection.

The main innovations of this work are summarized as follows:

- (1) A skeleton-weighted geometric guidance method is developed to estimate crack-propagation directions while reducing the influence of irregular boundaries and local branches, thereby providing an interpretable prior for active viewpoint planning.
- (2) A human-in-the-loop supervision pipeline is introduced to verify and correct the automatically generated directions, converting geometric priors into a task-specific dataset of expert-refined camera-motion labels.
- (3) An AdLoRA-based VLM adaptation strategy is developed for directional decision-making, combining circular supervision with hard-action-conditioned adversarial regularization of the continuous visual representation to improve robust active inspection.

The remainder of this paper is organized as follows. Section 2 reviews related work on MPA estimation, policy learning, and VLM adaptation. Section 3 presents the proposed framework, including Skeleton-Based Guidance, Human-in-the-loop Supervision, and AdLoRA for VLM. Section 4 reports the experimental results, including ablation studies, comparisons of direction prediction accuracy and inspection efficiency. Section 5 concludes the paper.

2. Related Work

2.1. Direction Estimation for Active Vision Inspection

Reliable direction estimation is a prerequisite for effective active vision inspection. Defect propagation direction is commonly characterized by MPA, whose estimation methods mainly include PCA-based and skeleton-based approaches. PCA-based methods compute the MPA from the spatial distribution of defect pixels, contours, or local descriptors by forming a covariance matrix and taking the dominant eigenvector as the MPA. Representative PCA-based methods include RPCA [8], cascading PCA [19], and LBP-PCA [20]. However, PCA-based methods remain sensitive to segmentation quality, noise, low contrast, branch interference, and irregular boundary fluctuations. On the other hand, Skeleton-based methods reduce the segmented defect region to a one-pixel-wide medial axis and then estimate the MPA from skeleton topology [9, 10, 21]. Although this method is more structurally interpretable, skeleton-based methods remain vulnerable to segmentation defects, unstable skeleton extraction, branch crossings, and fragmented centerlines. These limitations motivate methods that retain the interpretability of explicit direction estimation while improving robustness under complex conditions.

2.2. Learning-Based Active Vision Inspection

When explicit direction estimation is unreliable, learning-based methods provide a natural alternative. Recent studies model active vision as POMDP and train DRL agents to choose the next viewpoint based on the current observation and robot state [7, 22]. These methods can optimize long-term objectives related to segmentation quality and inspection efficiency, while reducing redundant full-area inspection compared with passive vision [5]. However, learning-based active vision inspection still faces several practical limitations. Policy learning typically requires large amounts of interaction data or simulation rollout, which makes training costly and slows adaptation to a new inspection environment. In addition, policy performance depends strongly on reward design, and poorly specified rewards can bias the agent toward short-term motion heuristics rather than reliable inspection behavior. Deploying DRL agents in industry also requires nontrivial integration of perception, environment modeling, and control, which increases system complexity and weakens transfer across materials, textures, lighting conditions, and defect morphologies. These limitations indicate that more data-efficient methods of active vision inspection remain necessary.

2.3. VLM Adaptation for Active Vision Inspection

Recently, VLM has emerged as a promising foundation for active vision inspection because of strong visual reasoning capabilities. Studies show that VLM can be adapted for embodied decision-making and navigation [23,24]. However, directly full-parameter fine-tuning VLM is often computationally expensive and difficult to sustain when task-specific data are limited. As a result, VLM adaptation typically relies on PEFT methods such as Low-Rank Adaptation (LoRA), which reduce the number of trainable parameters while preserving the capability of the pre-trained backbone.

However, standard LoRA remains sensitive to noise, occlusion, and domain shift. Adversarially enhanced LoRA improves robustness by incorporating perturbations, curriculum scheduling, or min-max optimization into training. In particular, AdvCLIP-LoRA [25], AdvLoRA [26], and DAC-LoRA [27] report improved robustness in VLM or few-shot adaptation. LoFT [28] and FullLoRA [29] further show that adversarially trained low-rank branches can approach the benefits of full adversarial fine-tuning with lower parameter cost. Yet these studies focus mainly on recognition or generic robustness benchmarks rather than directional decision-making for active vision inspection. This gap highlights the demand for robust VLM adaptation methods tailored to directional decision-making in active vision inspection.

Overall, three issues remain unresolved: existing MPA estimation methods are vulnerable to branch interference and local artifacts, learning-based active inspection remains costly to train and difficult to transfer, and standard LoRA remains insufficiently robust for industrial conditions. The next section addresses these issues with a framework that combines Skeleton-Based Guidance, Human-in-the-loop Supervision, and AdLoRA fine-tuning for active vision inspection.

3. Methodology

3.1. Task Formulation

This paper focuses on active crack inspection, which is formulated as a next best viewpoint (NBV) problem in which the system iteratively selects camera motions to improve crack visibility [30,31]. In the present setup, the NBV problem is simplified from continuous pose optimization to planar translation on the local camera-motion plane. The overall active vision inspection framework is shown in Figure 1.

Specifically, the action space is discretized into translation directions with 10° angular intervals. Given the current crack image, the VLM predicts a discrete label that indicates the next camera-motion direction:

$$\hat{y}_{k+1} = \arg \max_{y \in \mathcal{Y}} p_{\psi}(y | I_k), \quad |\mathcal{Y}| = K = 36 \quad (1)$$

where I_k is the current crack image, $\hat{y}_{k+1}$ is the predicted translation label, $\mathcal{Y}$ is the candidate direction set, K is the number of direction classes, and p_{ψ} is the VLM policy parameterized by ψ . Each label corresponds to one planar translation direction, and the complete inspection trajectory is generated by repeatedly executing these single-step NBV decisions.

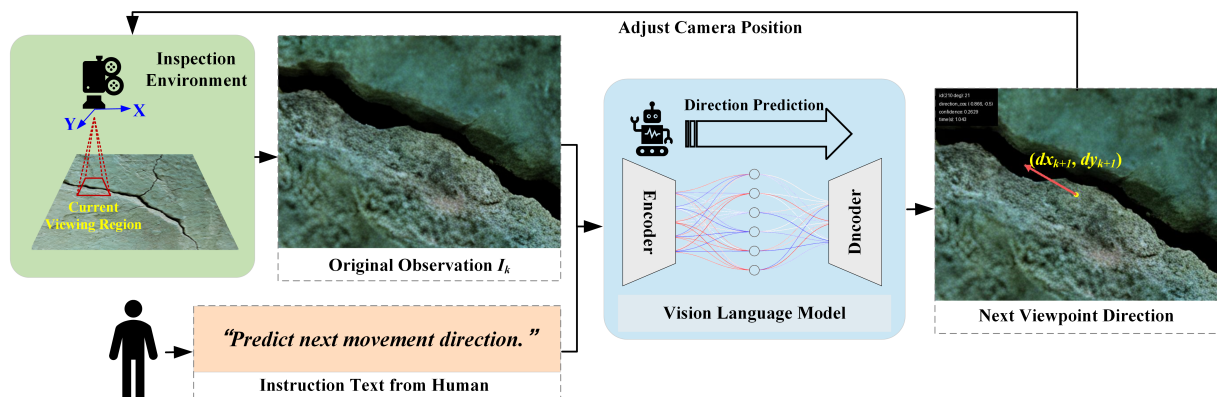


Figure 1. Active vision inspection framework. Based on the current observation, the VLM predicts the next motion direction so that the camera can move to an appropriate viewpoint, thereby improving the reliability of inspection decisions.

3.2. Human-in-the-Loop Supervision Pipeline for Dataset Collection

This subsection describes the human-in-the-loop supervision pipeline used to establish expert inspection priors. The raw images and masks are taken from the Ultralytics Crack Segmentation Dataset [32]. For each sample,

skeleton-weighted PCA provides an automatic directional prior, which is then refined through Human-in-the-Loop Supervision and exported in JSONL format.

3.2.1. Skeleton-Weighted PCA for Geometric Prior

Given a binary crack mask M and a structuring element B , let $M^{(t)}$ denote the mask obtained after t successive erosions by B . The residue used to construct the morphological skeleton at iteration t is

$$R^{(t)} = M^{(t)} \setminus (M^{(t)} \circ B) = M^{(t)} \setminus [(M^{(t)} \ominus B) \oplus B], \quad (2)$$

Here, $\ominus$, $\oplus$, and $\circ$ denote morphological erosion, dilation, and opening, respectively, with $A \circ B = (A \ominus B) \oplus B$. The erosion sequence is defined recursively below, where T denotes the index of its final nonempty mask:

$$\begin{aligned} M^{(0)} &= M, \\ M^{(t+1)} &= M^{(t)} \ominus B, \quad t = 0, 1, \dots, T-1, \\ T &= \max \{t \geq 0 : M^{(t)} \neq \emptyset\}. \end{aligned} \quad (3)$$

The morphological skeleton S is therefore the union of the residual sets $R^{(t)}$ for $t = 0, \dots, T$:

$$S = \text{Skel}(M) = \bigcup_{t=0}^T R^{(t)}. \quad (4)$$

These definitions explicitly distinguish erosion, opening, set difference, and union, thereby eliminating ambiguity in operator precedence. In the numerical implementation, iterative connectivity-preserving thinning is applied until convergence to obtain a one-pixel-wide centerline, which defines the skeleton set S used for distance weighting. The nearest-skeleton distance of each contour point is defined as

$$d_i = \min_{s \in S} \|p_i - s\|_2. \quad (5)$$

where p_i is the i th contour point of the crack region, N is the number of contour points, s is a skeleton point, and d_i is the distance from p_i to its nearest skeleton point. Instead of assigning equal weights to all contour points, a Gaussian distance-based weighting strategy is used so that points close to the skeleton centerline dominate the principal-axis estimate:

$$w_i = \exp\left(-\frac{d_i^2}{2\sigma^2}\right). \quad (6)$$

where w_i is the weight of contour point p_i , and σ controls the spatial decay of the skeleton-distance weighting. The weighted mean vector and weighted covariance matrix are computed as

$$\mu = \frac{\sum_{i=1}^N w_i p_i}{\sum_{i=1}^N w_i}, \quad \Sigma = \frac{\sum_{i=1}^N w_i (p_i - \mu)(p_i - \mu)^\top}{\sum_{i=1}^N w_i}. \quad (7)$$

where μ is the weighted mean vector and Σ is the weighted covariance matrix. At inspection step k , let Σ_k denote the weighted covariance matrix computed from the current crack contour. The unoriented principal crack axis u_k , defined as the unit eigenvector associated with the largest eigenvalue of Σ_k , is given by

$$u_k = \arg \max_{\|v\|_2=1} v^\top \Sigma_k v. \quad (8)$$

Because u_k and $-u_k$ define the same unoriented PCA axis, the sign ambiguity is resolved by enforcing temporal directional continuity. Let r_0 be the prescribed initial inspection direction and $r_k = v_{\text{wpca},k-1}$ for $k \geq 1$. The directed camera-motion vector is

$$v_{\text{wpca},k} = \arg \max_{v \in \{u_k, -u_k\}} v^\top r_k = \begin{cases} u_k, & u_k^\top r_k \geq 0, \\ -u_k, & u_k^\top r_k < 0. \end{cases} \quad (9)$$

Accordingly, maximizing $v^\top r_k$ selects the candidate closest in angle to the reference direction r_k , thereby preventing arbitrary 180° sign reversals between consecutive camera-motion decisions. Compared with standard PCA, this method suppresses local protrusions and irregular boundary artifacts while preserving useful branch

information when a branch remains close to the main trunk. The skeleton provides the structural reference, the distance term controls the contribution of each contour point, and the dominant eigenvector gives the predicted extension tendency of crack. The skeleton-weighted PCA algorithm for crack direction estimation is given in Algorithm 1.

Algorithm 1 Skeleton-weighted PCA

- 1: **Input:** Binary crack mask M
 - 2: Extract the largest connected region from M
 - 3: Thin the region to obtain the skeleton set S
 - 4: Extract the outer contour points $\{p_i\}_{i=1}^N$
 - 5: **for** $i = 1$ to N **do**
 - 6: Compute $d_i \leftarrow \min_{s \in S} \|p_i - s\|_2$
 - 7: Compute $w_i \leftarrow \exp(-d_i^2 / (2\sigma^2))$
 - 8: **end for**
 - 9: At inspection step k , compute the weighted mean μ_k and covariance matrix Σ_k , and then compute the unit principal eigenvector u_k of Σ_k
 - 10: Set r_0 to the prescribed initial inspection direction; for $k \geq 1$, set $r_k \leftarrow v_{\text{wpca}, k-1}$
 - 11: Set $v_{\text{wpca}, k}$ to the candidate in $\{u_k, -u_k\}$ that maximizes $v^\top r_k$
-

3.2.2. Human-in-the-Loop Direction Supervision

The algorithmic prior obtained from Algorithm 1 is written as

$$v_{\text{wpca}} = (d_x, d_y), \quad (10)$$

where (d_x, d_y) denotes the Cartesian components of v_{wpca} . Although v_{wpca} provides a useful geometric prior, it may not always match human inspection intent in cases such as branch crossings, weak contrast, or partial visibility. To support manual verification and correction, a human-in-the-loop direction supervision interface is used, as shown in Figure 2. If the predicted direction is acceptable, the algorithmic prior is exported directly. Otherwise, the annotator clicks a target point on the image plane to indicate the desired camera-motion direction, and the corresponding human direction is computed as

$$v_h = \frac{q - c}{\|q - c\|_2}. \quad (11)$$

where q is the clicked target point, c is the image center, and v_h is the human guidance direction. When the algorithmic prior does not match the intended inspection direction, the human guidance vector is introduced before sample export, especially for ambiguous samples affected by branch crossings, weak contrast, or partial crack visibility.



Figure 2. Human-in-the-loop direction supervision interface. The left panel supports click-based direction supervision, and the right panel previews JSONL records before export. The PASS button excludes from JSONL export any sample whose fused direction is ambiguous or unsuitable for supervision.

The geometric prior and human guidance are first combined into an intermediate vector v_{raw} . For a sample retained for supervision, v_{raw} is normalized only when it satisfies the following validity condition:

$$v_{\text{raw}} = v_{\text{wpca}} + \beta v_h, \quad v_f = \frac{v_{\text{raw}}}{\|v_{\text{raw}}\|_2}, \quad \|v_{\text{raw}}\|_2 > \epsilon, \quad (12)$$

where $\beta \in \{0, 1\}$ indicates whether human guidance is incorporated, v_f denotes the final fused unit vector, and $\epsilon > 0$ is a tolerance for detecting directional cancellation. When $\beta = 0$, $v_{\text{raw}} = v_{\text{wpca}}$, so the geometric prior is retained; when $\beta = 1$, $v_{\text{raw}} = v_{\text{wpca}} + v_h$, so human guidance contributes to the fused direction. The operator selects the PASS button shown in Figure 2 when the geometric and human-guidance vectors cancel or nearly cancel ($\|v_{\text{raw}}\|_2 \leq \epsilon$), or when the resulting direction is considered unsuitable for supervision. The corresponding sample is excluded before angle calculation, label assignment, and JSONL export. This expert-controlled rejection step prevents samples with ambiguous directions from entering the training set. The final vector of an accepted sample is then converted into an angle and quantized into the 36-direction action space:

$$\theta = \left(\text{atan2}(v_y, v_x) \cdot \frac{180^\circ}{\pi} + 360^\circ \right) \bmod 360^\circ, \quad y = \text{round} \left(\frac{\theta}{10^\circ} \right) \bmod K, \quad (13)$$

where (v_x, v_y) denotes the Cartesian components of v_f , $\theta \in [0, 360)$ is the corresponding motion angle, and y denotes the final discrete motion label. The human-in-the-loop supervision for generating JSONL is given in Algorithm 2.

Algorithm 2 Human-in-the-loop JSONL generation

- 1: **Input:** Crack image, bounding box, $v_{\text{wpca}} = (d_x, d_y)$
 - 2: Display the image, region overlay, and v_{wpca}
 - 3: **if** the displayed direction is acceptable **then**
 - 4: Set $\beta \leftarrow 0$ and initialize the human-guidance vector as $v_h \leftarrow (0, 0)$
 - 5: **else**
 - 6: Set $\beta \leftarrow 1$
 - 7: Acquire a click point q
 - 8: Compute $v_h \leftarrow (q - c) / \|q - c\|_2$
 - 9: **end if**
 - 10: Compute $v_{\text{raw}} \leftarrow v_{\text{wpca}} + \beta v_h$
 - 11: **if** PASS is selected or $\|v_{\text{raw}}\|_2 \leq \epsilon$ **then**
 - 12: Skip the current sample and continue
 - 13: **end if**
 - 14: Normalize v_{raw} to obtain $v_f \leftarrow v_{\text{raw}} / \|v_{\text{raw}}\|_2$
 - 15: Compute $\theta \leftarrow \text{atan2}(v_y, v_x)$ from $v_f = (v_x, v_y)$
 - 16: Compute $y \leftarrow \text{round}(\theta / 10^\circ) \bmod K$
 - 17: Build the instruction with the <bbox> field
 - 18: Save {image, instruction, label} to JSONL
-

Using the above human-in-the-loop supervision procedure, we constructed a finalized JSONL dataset containing 2467 samples, which was divided into training (2048; 83.02%), validation (209; 8.47%), and test (210; 8.51%) sets. The human-intervention rate was 48.50%, and the average annotation time was 2.71 s. During data collection, each sample was categorized as Accept, Correct, or Pass based on the outcome of human verification; representative examples are shown in Figure 3.

To focus the VLM on the crack region, the crack bounding box is injected into the training prompt through a dedicated <bbox> field. Each accepted or corrected sample is exported as a JSONL record with the image identifier, an instruction containing the <bbox> token, and the final label. The <bbox> field is embedded directly into the multimodal instruction so that the model focuses direction prediction on the crack-centered region during fine-tuning. A JSONL sample is as follows:

```
{
  "image": "crack-001.jpg",
  "instruction": "The crack is located at <bbox>100 80 200 180</bbox>. Predict next movement direction.",
  "label": 4
}
```

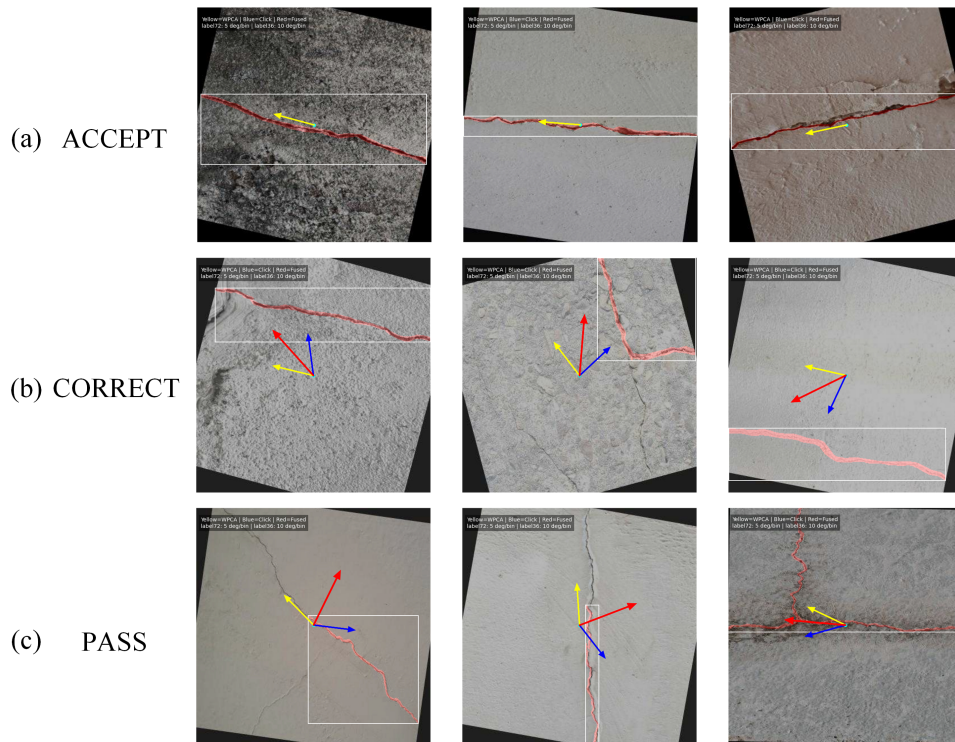


Figure 3. Representative outcomes of the human-in-the-loop annotation audit: (a) ACCEPT retains the automatically estimated direction without click-based correction; (b) CORRECT replaces an unsuitable automatic estimate with an annotator-corrected fused direction; and (c) PASS excludes an ambiguous or unusable sample from export.

3.3. AdLoRA Training

3.3.1. Model Backbone and State Representation

This subsection defines the model backbone and the state representation used for AdLoRA training. The overall AdLoRA architecture is shown in Figure 4. AdLoRA training uses the finalized image–instruction–action samples constructed in Section 3.2. The policy backbone is Qwen2.5-VL-3B-Instruct [33]. LoRA adapters are inserted only into the attention projection matrices q_{proj} and v_{proj} , while the vision encoder, multimodal projector, and language backbone remain frozen. The trainable and frozen modules are summarized in Table 1. During data loading, the processor wraps each sample into a chat-style multimodal prompt and converts it into token IDs, attention masks, visual patches, and image-grid metadata compatible with Qwen2.5-VL.

Table 1. Trainable and frozen modules in AdLoRA.

Module	Status	Params.
Vision encoder	Frozen	0
Multimodal projector	Frozen	0
LLM backbone	Frozen	0
LoRA on q_{proj}	Trainable	≈2.36 M
LoRA on v_{proj}	Trainable	≈2.36 M
Linear classifier head	Trainable	≈73.8 K
State–action discriminator	Trainable	≈1.20 M
Total trainable		≈6.0 M

The encoded hidden sequence is written as

$$H = \Phi(I, t), \quad H \in \mathbb{R}^{L \times d_h}, \quad (14)$$

where $\Phi(\cdot)$ denotes the multimodal backbone with LoRA adapters, I is the RGB crack image, t is the multimodal instruction containing the $\langle \text{box} \rangle$ -based crack-region prompt, H is the encoded hidden sequence, L is the sequence length, and d_h is the hidden dimension.

The last valid token is used as the pooled state feature:

$$h = H_\ell, \quad (15)$$

where h is the pooled state feature and ℓ is the index of the last valid token. The pooled feature is then passed to a linear classification head to produce the direction logits:

$$z = Wh + b, \quad z \in \mathbb{R}^K. \quad (16)$$

where z is the direction-logit vector, and W and b are the weight matrix and bias vector of the linear classification head, respectively. The logits are then normalized by the softmax function to obtain the policy distribution:

$$\pi(k | I, t) = \text{softmax}(z)_k, \quad k \in \{0, \dots, K-1\}. \quad (17)$$

where $\pi(k | I, t)$ denotes the predicted probability of direction class k . During adversarial training, the hard policy action is selected by applying $\arg \max$ to the logits, and its one-hot encoding is treated as a non-differentiable conditioning variable. Let $\text{sg}(\cdot)$ denote the stop-gradient operator and $e(\cdot)$ denote one-hot encoding. The inputs used for the discriminator and policy updates are therefore defined separately as

$$\begin{aligned} a_\pi &= \arg \max_k z_k, & \tilde{e}_\pi &= \text{sg}(e(a_\pi)), \\ x_D^E &= [\text{sg}(h); e(y)], & x_D^\pi &= [\text{sg}(h); \tilde{e}_\pi], \\ x_P^\pi &= [h; \tilde{e}_\pi]. \end{aligned} \quad (18)$$

During the discriminator update, h and $\tilde{e}_\pi$ are detached from the policy computation graph. During the policy update, h remains connected to the LoRA-adapted backbone, whereas $\tilde{e}_\pi$ remains detached. The discriminator maps each state–action input to a scalar logit:

$$D(x) \in \mathbb{R}, \quad x \in \{x_D^E, x_D^\pi, x_P^\pi\}. \quad (19)$$

Thus, gradients do not propagate through either $\arg \max$ or the one-hot encoding.

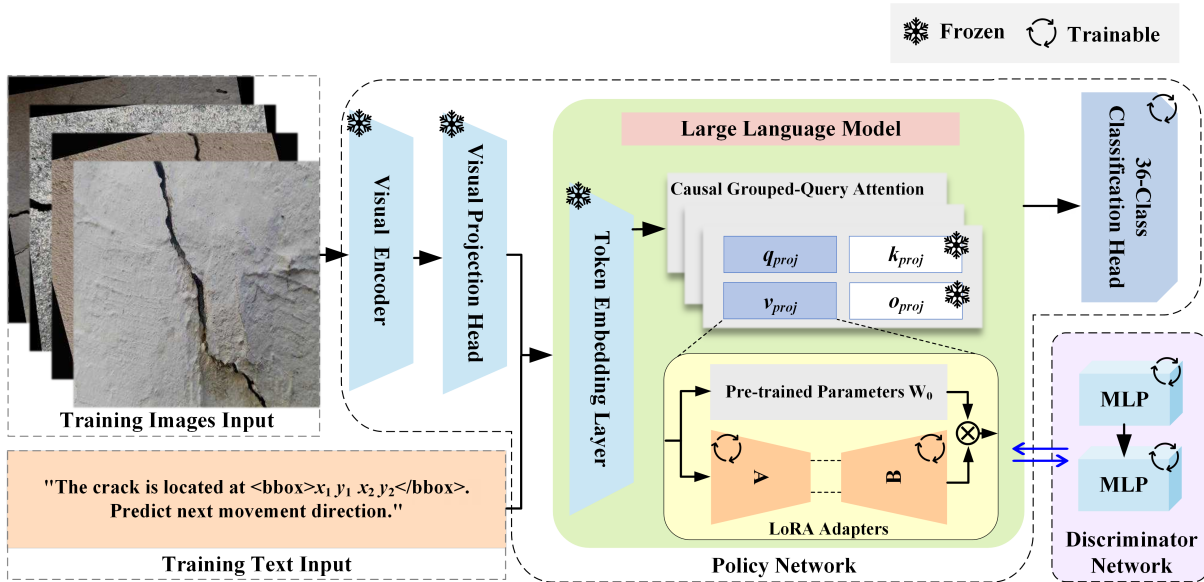


Figure 4. AdLoRA architecture. The image and instruction are encoded by a shared Qwen2.5-VL backbone, in which the visual encoder, multimodal projector, and language backbone remain frozen, while LoRA adapters are trained on the q_proj and v_proj attention layers. The pooled multimodal representation is then fed to a trainable direction classifier and a trainable discriminator for policy learning and adversarial training.

3.3.2. Training Objective

Because the camera-motion labels lie in a circular direction space, ordinary one-hot cross-entropy would over-penalize predictions that differ from the reference by only one neighboring direction. The training objective combines a supervised circular classification loss, a discriminator loss, an adversarial alignment term, and an entropy regularizer. The supervised term is defined as circular cross-entropy (CCE) with a circularly smoothed target

distribution. Under the training configuration adopted in this study, the remaining probability mass is uniformly assigned to the two adjacent labels with

$$\omega_n = \frac{1 - \omega_c}{2\tau}. \quad (20)$$

where τ denotes the circular tolerance, ω_c denotes the probability mass assigned to the center label, and ω_n denotes the probability mass assigned to each neighboring label. In this study, $\tau = 1$ and $\omega_c = 0.85$.

The target distribution centered at y is defined as

$$p^*(k | y) = \begin{cases} \omega_c, & k = y, \\ \omega_n, & k \in \{y \pm 1, \dots, y \pm \tau\} \bmod K, \\ 0, & \text{otherwise,} \end{cases} \quad (21)$$

where $p^*(k | y)$ is the smoothed target probability for direction class k centered at the reference label y . Based on this smoothed circular target distribution, the CCE loss is defined as

$$\mathcal{L}_{\text{circ}} = - \sum_{k=0}^{K-1} p^*(k | y) \log \pi(k | I, t). \quad (22)$$

where $\mathcal{L}_{\text{circ}}$ is the circular cross-entropy loss. Using the detached inputs x_D^E and x_D^π , the discriminator is optimized with the loss

$$\mathcal{L}_D = \text{BCE}(D(x_D^E), 1) + \text{BCE}(D(x_D^\pi), 0). \quad (23)$$

where $\mathcal{L}_D$ is the discriminator loss and $\text{BCE}(\cdot)$ denotes the binary cross-entropy (BCE) loss. During the policy update, the discriminator parameters are held fixed and $\tilde{e}_\pi$ remains detached, whereas h remains differentiable. The adversarial loss is

$$\mathcal{L}_{\text{adv}} = \text{BCE}(D(x_P^\pi), 1). \quad (24)$$

Accordingly, the only differentiable path from $\mathcal{L}_{\text{adv}}$ to the LoRA parameters φ passes through the continuous feature h_φ :

$$\nabla_\varphi \mathcal{L}_{\text{adv}} = \frac{\partial \mathcal{L}_{\text{adv}}}{\partial h_\varphi} \frac{\partial h_\varphi}{\partial \varphi}, \quad \frac{\partial \mathcal{L}_{\text{adv}}}{\partial \tilde{e}_\pi} = 0.$$

Therefore, $\mathcal{L}_{\text{adv}}$ regularizes the continuous LoRA representation while being conditioned on a detached hard action; it does not propagate gradients through the discrete action. The circular loss and entropy term directly optimize the classification head. To avoid overly peaked predictions, the policy is further regularized by the entropy term

$$\mathcal{H}(\pi) = - \sum_{k=0}^{K-1} \pi(k | I, t) \log \pi(k | I, t). \quad (25)$$

where $\mathcal{H}(\pi)$ is the policy entropy. The final policy objective is therefore written as

$$\mathcal{L}_{\text{policy}} = \lambda_{\text{sup}} \mathcal{L}_{\text{circ}} + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}} - \lambda_{\text{ent}} \mathcal{H}(\pi), \quad (26)$$

where $\mathcal{L}_{\text{policy}}$ is the final policy loss, and λ_{sup} , λ_{adv} , and λ_{ent} are the weights of the supervised, adversarial, and entropy terms, respectively. The adopted setting uses $\lambda_{\text{sup}} = 1.0$ and $\lambda_{\text{ent}} = 5 \times 10^{-5}$. For the two-stage adversarial schedule, $\lambda_{\text{adv}} = 0$ in the first stage and $\lambda_{\text{adv}} = 0.01$ in the second stage. The policy therefore first learns a stable supervised direction classifier and then applies hard-action-conditioned adversarial alignment to its continuous LoRA-adapted representation.

3.3.3. Training Protocol and Implementation Details

The training set contains 2048 image–instruction–action samples, and the validation set contains 209 samples. The same 2048 training samples are first used for discriminator warm-up and then reused for joint adversarial optimization, rather than being split into separate discriminator-only and policy-only subsets. To reduce the influence of class imbalance, the training loader uses class-balanced sampling. During discriminator updates, the policy forward pass is evaluated without gradient tracking, and the resulting argmax action and one-hot encoding serve

only as detached discriminator inputs. During policy updates, the discriminator optimizer is not stepped; the hard action remains detached, while the adversarial gradient propagates through the continuous pooled feature h to the LoRA parameters φ . The circular cross-entropy and entropy terms directly optimize the classification head.

The optimization follows an alternating update strategy. The discriminator is first warmed up for two epochs on expert and policy-induced state–action pairs, while the policy parameters remain fixed. After warm-up, joint training alternates discriminator and policy updates within each mini-batch. The first adversarial stage lasts for one epoch with zero adversarial weight, and the second stage activates adversarial learning with $\lambda_{\text{adv}} = 0.01$. This schedule reduces early training instability and helps the classifier establish a reliable supervised prior before adversarial alignment is imposed.

The key training hyperparameters are summarized in Table 2, and the training curves are shown in Figure 5. The feasibility experiment was conducted on a single NVIDIA RTX PRO 6000 Blackwell Workstation Edition GPU with 96 GB of device memory. This moderate initial configuration was used to assess the convergence stability of the proposed alternating optimization and does not imply that this hyperparameter set is optimal. Specifically, $r = 16$ and $\alpha = 32$ yield an adapter scaling ratio of $\alpha/r = 2$, while a dropout of 0.05 regularizes the adapters for the limited task-specific dataset. The policy learning rate of 1×10^{-4} supports efficient adapter optimization, whereas the lower discriminator learning rate of 5×10^{-6} reduces the risk that the discriminator dominates the policy. A per-device batch size of 2 was used to fit the VLM and discriminator within GPU memory. Eight gradient-accumulation steps yielded an effective batch size of 16, and training was conducted for 12 epochs to assess convergence.

Table 2. AdLoRA training setup.

Param.	Value
LoRA configuration	$r = 16, \alpha = 32, \text{dropout} = 0.05$
Learning rate	1×10^{-4}
Discriminator learning rate	5×10^{-6}
Epochs	12
Batch size	2
Gradient accumulation steps	8
Circular loss setting	$\tau = 1, \omega_c = 0.85$
Validation criterion	± 1 label tolerance

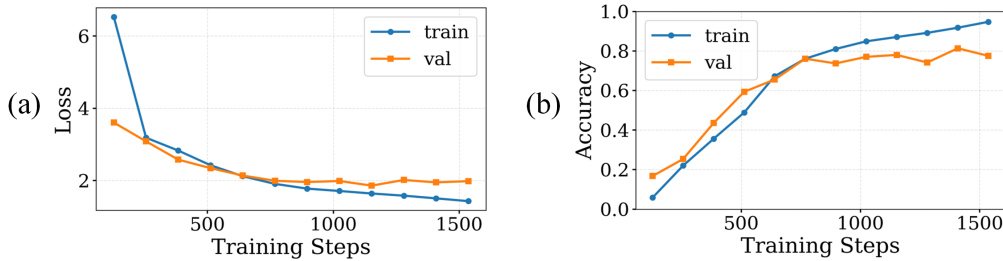


Figure 5. AdLoRA training curves: (a) training and validation CCE loss and (b) training and validation accuracy within the ± 1 -bin tolerance. The blue and orange curves denote the training and validation results, respectively.

The validation accuracy curve uses a tolerant criterion that counts a prediction as correct when its circular label distance to the ground truth does not exceed one neighboring direction. This criterion is physically reasonable because a one-step deviation in camera motion usually does not materially degrade downstream visibility. The best checkpoint is selected according to validation performance, yielding a best validation loss of 1.8593 and a peak validation accuracy of 0.8134. These trends indicate stable convergence and show that the proposed AdLoRA framework can learn an effective direction policy from the human-in-the-loop expert inspection priors. This feasibility configuration is denoted as group A in Section 4.1. The subsequent hyperparameter search and configuration selection for the formal method comparison are performed exclusively on the validation set.

The complete group-A AdLoRA training run took 1.48 h on the single GPU described above. By comparison, the DRL-based active vision inspection study in [7] used 20,000 training iterations and approximately 3.6 million experience steps for a similar task. Normalizing this interaction budget to our measured simulator throughput yields an estimated training time of 143.7 h. This comparison indicates a training-efficiency advantage for the proposed method, which updates only low-rank adapters and lightweight task heads of a pre-trained VLM, whereas the DRL approach learns a task-specific policy through environment interaction.

4. Experiments

4.1. Ablation Study

This ablation study evaluates the effect of adversarial training on directional prediction by comparing LoRA and AdLoRA across five hyperparameter groups, denoted A–E. The group settings are listed in Table 3. Within each group, the two methods share the same hyperparameter setting, training/validation split, 36-class action space, and CCE objective defined in Section 3.3. Two metrics are reported for all A–E experiments: validation CCE loss and validation accuracy. Group A corresponds to the initial feasibility configuration in Section 3.3.3, whereas groups B–E evaluate alternative combinations of learning rate, number of training epochs, LoRA rank and scaling factor, dropout, and weight decay within the available computational budget.

Table 3. Hyperparameter settings of groups A–E.

Group	Learning Rate	Epoch	Rank	Alpha	Dropout	Weight Decay
A	1×10^{-4}	12	16	32	0.05	–
B	5×10^{-5}	15	32	64	0.05	–
C	3×10^{-5}	12	32	64	0.10	0.02
D	2×10^{-5}	15	16	32	0.10	0.05
E	5×10^{-5}	8	32	64	0.08	0.01

The quantitative comparison is summarized in Table 4, while Figure 6 shows the corresponding validation trends for the matched A–E groups: panel (a) presents the CCE loss curves, and panel (b) presents the accuracy curves. In Table 4, the reported CCE value for each group is the minimum validation loss reached during training, whereas the reported accuracy is the maximum validation accuracy achieved under the same hyperparameter setting. The differences are calculated relative to LoRA; therefore, a negative ΔCCE and a positive $\Delta\text{Accuracy}$ both indicate an improvement obtained with AdLoRA.

Table 4. Performance of groups A–E.

Group	LoRA CCE	AdLoRA CCE	ΔCCE	LoRA Accuracy	AdLoRA Accuracy	$\Delta\text{Accuracy}$
A	2.0425	1.8593	−0.1832	0.7081	0.8134	+0.1053
B	1.8535	1.8202	−0.0333	0.8230	0.8373	+0.0143
C	1.9248	1.8018	−0.1230	0.7895	0.8469	+0.0574
D	2.0371	1.9921	−0.0450	0.6890	0.7464	+0.0574
E	1.9417	1.8038	−0.1380	0.7608	0.8230	+0.0622

Across all five matched hyperparameter groups, AdLoRA improves both evaluation metrics simultaneously. It reduces the minimum validation CCE loss in every group, with an average reduction of 0.1045 and the largest reduction of 0.1832 in group A. This consistent reduction suggests that adversarial optimization improves alignment with the circular target distribution itself, rather than merely accelerating convergence under one particular configuration. Under the ± 1 -bin tolerance criterion, AdLoRA also achieves higher maximum validation accuracy in every group, with an average gain of 0.0593. The largest accuracy gain is obtained in group A, increasing from 0.7081 to 0.8134, whereas group C achieves the highest overall AdLoRA accuracy of 0.8469, compared with 0.7895 for LoRA under the same setting. Even in group B, where the improvements are comparatively modest, both metrics still change in the favorable direction. This consistency across different learning rates, training epochs, ranks, dropout rates, and weight-decay settings indicates that the benefit is associated with the adversarial optimization strategy rather than with a single favorable hyperparameter choice. Because neighboring direction bins correspond to only small camera-motion deviations, the accuracy improvements further indicate more reliable motion decisions for practical inspection. Group C attains the lowest AdLoRA validation CCE of 1.8018 and the highest AdLoRA validation accuracy; it is therefore selected for the formal prediction comparison in Section 4.2 without consulting the held-out test results.

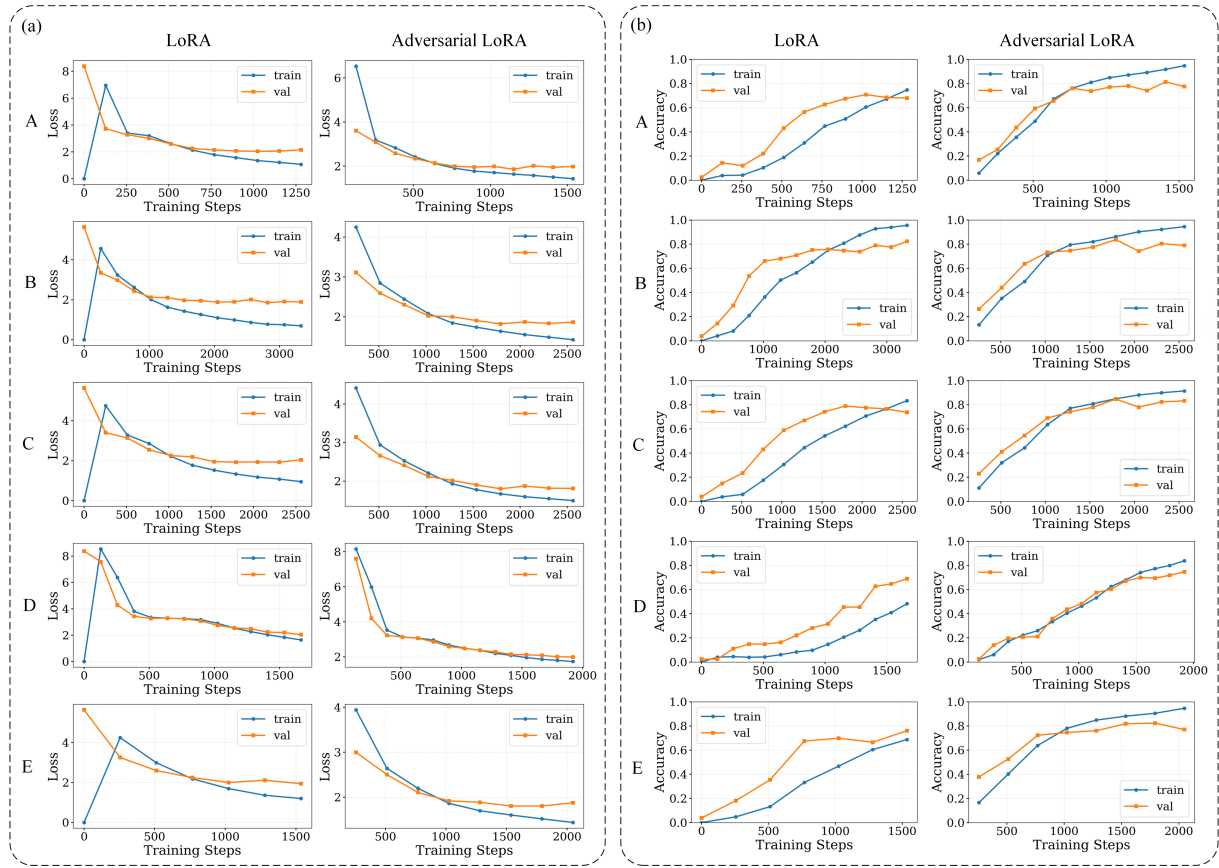


Figure 6. Ablation curves for hyperparameter groups (A)–(E) defined in Table 3: (a) CCE loss curves and (b) accuracy curves. In each panel, the five rows correspond to groups A–E, respectively, with LoRA shown in the left column and Adversarial LoRA shown in the right column. The blue and orange curves denote training and validation results, respectively. Adversarial LoRA consistently achieves lower validation CCE loss and higher maximum validation accuracy than LoRA across the matched hyperparameter groups.

4.2. Prediction Error Comparison

This subsection compares three inference pipelines on the 210-sample test split to quantify directional prediction errors under the same evaluation protocol. The compared pipelines are a VLM-based *linear probe* baseline, LoRA fine-tuning, and the proposed AdLoRA strategy. For a controlled comparison, LoRA and AdLoRA use the same group-C hyperparameters, which were selected exclusively from the validation results in Section 4.1. Group C achieved the lowest AdLoRA validation CCE and the highest validation accuracy within ± 1 bin, and the held-out test set was not consulted during selection. Under this protocol, the linear probe takes normalized multimodal embeddings as input and adopts a linear classifier projecting 2048-dimensional latent features into 36 directional categories, while LoRA and AdLoRA directly fine-tune the Qwen2.5-VL backbone for the 36-class directional prediction task. All three methods are evaluated with the same instruction format, crack bounding-box prompts, and 210-sample test split.

Three complementary metrics are reported. The circular label distance is defined as

$$d_{\text{circ}}(y, \hat{y}) = \min(|y - \hat{y}|, K - |y - \hat{y}|). \quad (27)$$

where $d_{\text{circ}}(y, \hat{y})$ is the shortest circular distance between the reference and predicted direction labels. The first metric is accuracy within ± 1 bin. In practical active vision, a small directional deviation can still move the camera toward a valid viewpoint for inspection. Therefore, this study allows a one-bin tolerance around the reference direction and counts a prediction as correct if $d_{\text{circ}}(y, \hat{y}) \leq 1$:

$$\text{Accuracy within } \pm 1 \text{ bin} = \frac{1}{N} \sum_{i=1}^N c_i. \quad (28)$$

where N is the number of test samples and $c_i = 1$ if $d_{\text{circ}}(y_i, \hat{y}_i) \leq 1$, and $c_i = 0$ otherwise. The second metric is the mean absolute angular error (MAAE), for which the single-sample circular angular error is defined as follows:

$$e_i = \min \left(|\theta_i - \hat{\theta}_i|, 360^\circ - |\theta_i - \hat{\theta}_i| \right). \quad (29)$$

where e_i is the circular angular error of sample i , $\theta_i = 10^\circ y_i$ is the reference angle, and $\hat{\theta}_i = 10^\circ \hat{y}_i$ is the predicted angle. The MAAE is then computed as

$$\text{MAAE} = \frac{1}{N} \sum_{i=1}^N e_i. \quad (30)$$

The third metric is the test-set CCE loss. Following the circular smoothing strategy used in training, the evaluation target distribution is defined as

$$p^*(k | y) = \begin{cases} \omega_c, & k = y, \\ \frac{1-\omega_c}{2\tau}, & k \in \{y \pm 1\} \bmod K, \\ 0, & \text{otherwise,} \end{cases} \quad (31)$$

where the smoothing parameters follow the setting defined in Section 3.3. Based on the same smoothed target distribution, the third metric is the test-set CCE loss:

$$\mathcal{L}_{\text{CCE}} = -\frac{1}{N} \sum_{i=1}^N \sum_{k=0}^{K-1} p^*(k | y_i) \log \pi_i(k). \quad (32)$$

where $\mathcal{L}_{\text{CCE}}$ is the test-set CCE loss and $\pi_i(k)$ is the predicted probability of class k for sample i . Lower values indicate that the predicted distribution is better aligned with the smoothed circular target distribution.

As shown in Figure 7, the linear-probe baseline achieves only 0.229 in accuracy within ± 1 bin, together with an MAAE of 52.620° and a CCE loss of 3.355, indicating that frozen multimodal features combined with a shallow classifier are insufficient for reliable directional decision-making in this task. LoRA fine-tuning substantially improves performance, reaching 0.757 in accuracy within ± 1 bin, 19.480° in MAAE, and 2.181 in CCE loss. AdLoRA further improves these metrics to 0.852, 14.240° , and 1.719, respectively. Together, these gains indicate better tolerance-aware directional correctness, higher angular precision, and stronger alignment with the circular target distribution. The consistent improvement across all three metrics shows that adversarial enhancement improves parameter-efficient VLM adaptation for direction prediction in active vision.

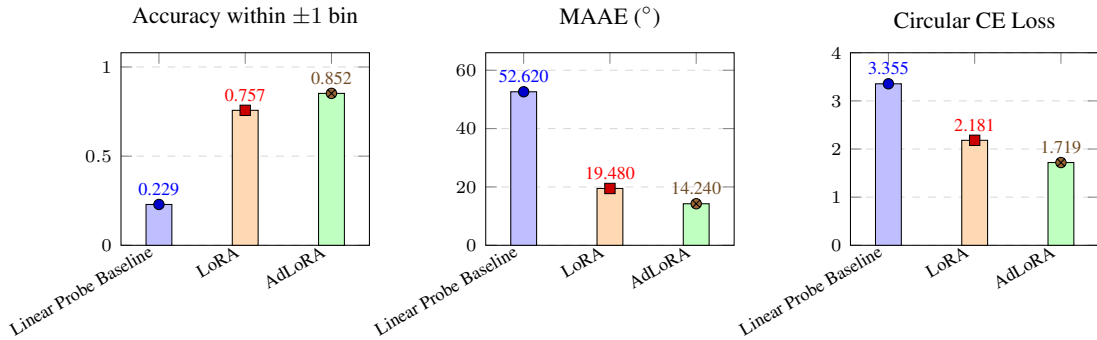


Figure 7. Prediction error comparison. The three figures show that AdLoRA achieves the best overall performance, with higher accuracy within ± 1 bin and lower MAAE and CCE loss than the compared methods.

To isolate the effects of geometric label construction and human supervision, three Qwen2.5-VL model instances were fine-tuned with AdLoRA using the group-C configuration. All three runs used the same 2048 training images and corresponding instructions, as well as identical 209-sample validation and 210-sample test sets. They differed only in the training-labeling strategy: skeleton-based estimation, skeleton-weighted PCA, or skeleton-weighted PCA with Human-in-the-Loop Supervision.

Table 5 shows that, relative to skeleton-weighted PCA, the proposed method increases the accuracy within ± 1 bin from 0.1857 to 0.8476 and reduces the MAAE from 102.86° to 14.29° . Accounting for the morphological crack skeleton provides a modest improvement in directional accuracy. However, because both PCA-based and skeleton-based estimators derive closely related geometric priors from the same crack mask, the gain from this change alone remains limited. In contrast, human-in-the-loop supervision substantially improves prediction accuracy by preventing unsuitable automatically generated directions from being used as training labels.

Table 5. Effects of skeleton-based geometric guidance and human-in-the-loop supervision

Training-Label Source	Accuracy within ± 1 Bin	MAAE ($^{\circ}$)
Skeleton-based estimation	0.1762	104.57
Skeleton-weighted PCA	0.1857	102.86
Proposed method	0.8476	14.29

To quantify run-to-run variability, the group-C AdLoRA configuration was trained independently using random seeds 42–46. All runs used the same training, validation, and test splits, and the best checkpoint from each run was selected according to validation CCE before evaluation on the 210-sample test set. Table 6 reports the sample mean and sample standard deviation (SD) across the five runs.

Table 6. Performance across five random-seed training runs using group-C hyperparameters

Metric	Mean	Sample SD
Accuracy within ± 1 bin	0.7981	0.0338
MAAE	15.3810 $^{\circ}$	1.2210 $^{\circ}$
CCE loss	1.7842	0.0436

4.3. Inspection Efficiency Comparison

4.3.1. Experimental Setup

For the efficiency evaluation in simulation, a crack inspection environment is built in MuJoCo, as shown in Figure 8. The setup uses a Universal Robots 5 (UR5) robot with an RGB camera mounted on the end effector. To facilitate intuitive monitoring and interactive operation, the interface offers comprehensive visualization and control capabilities, including real-time rendering of the current scene viewpoint, dynamic visualization of historical camera trajectories, on-site directional inference for camera orientation decision-making, and human-in-the-loop motion control that combines automatic policy output with manual manipulation.

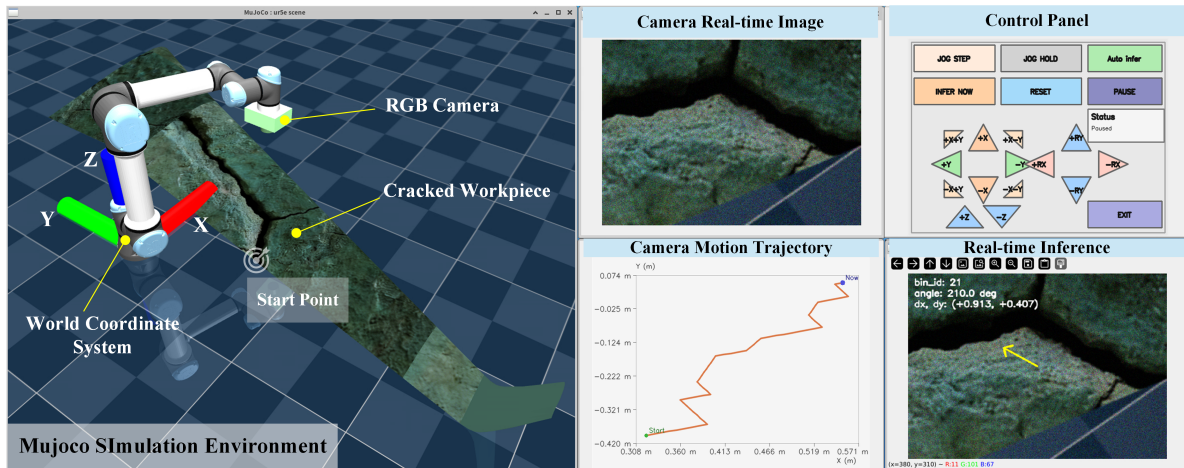


Figure 8. MuJoCo simulation environment. The setup includes a UR5 robot, an RGB camera, and a crack workpiece. The interface provides real-time viewpoint display, trajectory visualization, real-time direction inference, and human-in-the-loop motion control. The real-time camera and inference panels show observations degraded by spatially nonuniform illumination, Gaussian blur, Poisson shot noise, and Gaussian read noise, providing a reproducible stress test under low-contrast and noisy imaging conditions.

The comparison includes the proposed active vision method, a passive full-area inspection strategy, and a non-VLM active vision baseline based on PCA. Using the notation $v_f = (v_x, v_y)$ defined in Section 3.2.2, the predicted direction label is mapped to the image-plane unit vector $\hat{v}_f = (\hat{v}_x, \hat{v}_y)$, with the positive x -axis pointing right and the positive y -axis pointing downward. The controller first forms the camera-frame vector $\mathbf{u}_C = [\hat{v}_x, -\hat{v}_y, 0]^T$. Let ${}^B\mathbf{R}_C$ denote the current rotation matrix from the camera frame C to the robot-base frame B . The transformed vector $\mathbf{u}_B = {}^B\mathbf{R}_C\mathbf{u}_C$ is projected onto the base-frame XY plane and normalized. Let $\hat{\mathbf{u}}_{B,XY}$ denote the normalized planar direction. The commanded translation is $\Delta\mathbf{p}_B = (0.01 \text{ m})\hat{\mathbf{u}}_{B,XY}$, with the

end-effector height and orientation held fixed. The target pose is executed only if inverse kinematics admits a feasible solution. The same step size was used for all methods to balance viewpoint-adjustment resolution against trajectory length.

All methods are evaluated using the same initial joint configuration, inverse-kinematics constraints, workpiece pose, and visual conditions. Camera observations are generated under a fixed spatially nonuniform illumination pattern containing both bright and dark regions. A reproducible sensor-degradation pipeline is then applied, comprising a 9×9 Gaussian blur with $\sigma = 2.0$ pixels, Poisson shot noise with a full-scale photon count of 30, and Gaussian read noise with a normalized standard deviation of 0.025; all stochastic operations use random seed 42. The inspection terminates when the crack endpoint appears within the camera field of view. Inspection efficiency is quantified by the number of successfully executed camera-motion steps.

The active visual inspection task considered in this study is non-contact and does not require force exchange or dynamic interaction between the robot and the environment. Consequently, the inability of simulation to fully reproduce real contact dynamics has limited influence on the visual decision-making problem evaluated here. The simulated crack instances preserve the task-relevant physical attributes observable by the camera, including irregular morphology, orientation, scale, branching, and local contrast, while MuJoCo models robot kinematics, camera viewpoints, inverse-kinematics feasibility, and workspace constraints. Spatially nonuniform illumination, a 9×9 Gaussian blur, Poisson shot noise, and Gaussian read noise are further introduced to reproduce adverse imaging conditions. Together, these settings provide a credible approximation for assessing the relevant visual effects and motion feasibility.

4.3.2. Comparison Methods and Results

At each step, the proposed method uses the AdLoRA-adapted VLM to predict the next motion direction from the current camera observation and executes the corresponding planar motion if it is feasible. The passive vision baseline follows a predefined U-shaped trajectory covering the entire workpiece. The PCA-based method extracts candidate crack pixels by applying a fixed threshold $V \leq T$ to the HSV value channel, where $V \in [0, 255]$ and T is the manually selected dark-intensity threshold. Morphological opening and closing use 3×3 and 9×9 kernels, respectively. The largest connected component with an area of at least 80 pixels is retained, and its principal axis is estimated using unweighted PCA. Directional continuity resolves the sign ambiguity, after which the direction is quantized into the same action space as the VLM.

The comparison is summarized in Figure 9. Because all methods use the same motion step size and termination condition, the number of successfully executed camera motions provides a direct measure of inspection efficiency under the evaluated conditions. The proposed method completed the inspection in 93 steps, compared with 125 steps for the passive vision baseline. It therefore avoided 32 camera motions and reduced the executed step count by 25.6%. The passive strategy follows a predefined U-shaped path to cover the full workpiece, so it continues to traverse regions that are not required once the crack-propagation direction can be inferred. By contrast, the proposed method updates the motion direction from each camera observation and consequently reduces redundant workspace coverage while maintaining closed-loop viewpoint adjustment.

With $T = 40$, the PCA baseline completed the inspection in 89 steps, four fewer than the proposed method. This result shows that a simple geometric estimator can be highly step-efficient when the fixed threshold separates the crack from its background reliably. However, the successful trajectory depends on the quality of the segmented crack pixels. Increasing only the threshold to $T = 50$ admitted dark non-crack regions into the candidate mask, changed the dominant connected structure, and diverted the PCA direction from the actual crack. Because each estimated direction determines the next camera position, this local segmentation error also changes subsequent observations and can accumulate during closed-loop motion. Consequently, the PCA baseline reached the boundary of the reachable workspace and failed after 121 recorded steps rather than completing the inspection. The two PCA trials therefore demonstrate that a low step count in a successful run does not by itself establish robust inspection performance. In comparison, the proposed method completed the task under the same degraded imaging conditions without relying on a manually selected intensity threshold. Overall, the results indicate that the PCA baseline can be slightly more step-efficient when segmentation is reliable, whereas the proposed method provides a more favorable balance between completion reliability and motion efficiency; it retains the adaptive benefit of active vision while requiring substantially fewer steps than passive full-area coverage.

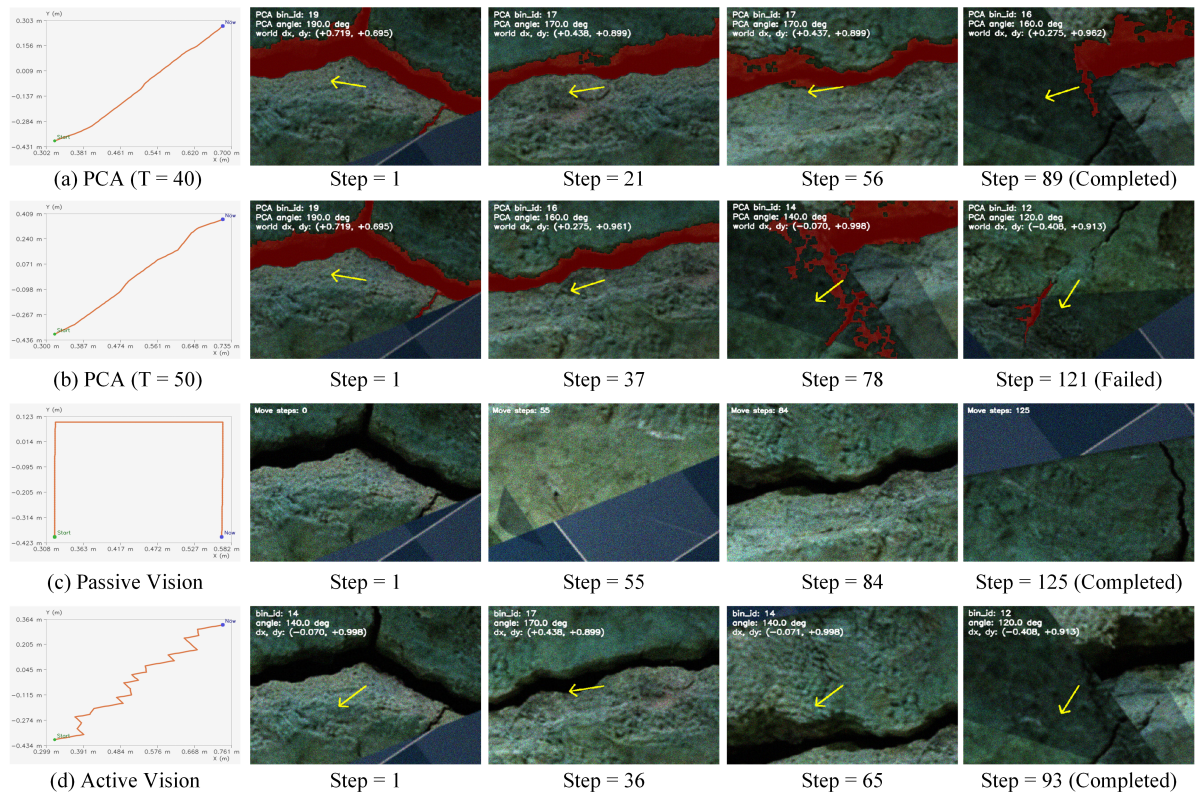


Figure 9. Comparison of inspection efficiency among four methods: (a) PCA with $T = 40$, which completed the inspection in 89 steps; (b) PCA with $T = 50$, which misclassified dark regions as cracks, reached the boundary of the reachable workspace, and failed after 121 steps; (c) passive vision, which completed the inspection in 125 steps; and (d) the proposed active vision method, which completed the inspection in 93 steps. In each row, the first panel shows the inspection trajectory, and the remaining four panels present representative camera observations at selected inspection steps.

5. Conclusions

The main contribution of this work is a unified active vision inspection framework that integrates skeleton-weighted geometric guidance, human-refined direction labels, and adversarial low-rank adaptation of a VLM. The ablation study shows that AdLoRA consistently outperforms standard LoRA under matched hyperparameter settings. The directional prediction comparison further shows that the proposed method achieves the best overall performance among the compared methods in terms of accuracy within ± 1 bin, MAAE, and CCE loss. The simulation shows that the proposed method requires fewer steps than the passive vision method, representing a 25.6% improvement in inspection efficiency. Future work will validate the framework using a physical robot and an on-site vision system.

Author Contributions

C.H.: conceptualization, methodology, software, formal analysis, investigation, data curation, visualization, and writing—original draft preparation; R.Q.: validation, investigation, and writing—reviewing and editing; L.X.: conceptualization, supervision, project administration, and writing—reviewing and editing; P.Z.: supervision, project administration, and writing—reviewing and editing. All authors have read and agreed to the published version of the manuscript.

Funding

This work was mainly supported by the funding support from the National Natural Science Foundation of China (No. 52422514), the National Centre of Technology Innovation for Intelligent Design and Numerical Control (NCDC)—Hong Kong Centre (PolyU P0060903), the General Research Fund (GRF) (Project No. PolyU 15210222 and PolyU15206723) and the Collaborative Research Fund (CRF) (Project No. C6044-23GF) from the Research Grants Council (RGC), Hong Kong, Shenzhen Fundamental Research Scheme-General Program (JCYJ20230807140407016), and the Guangdong-Hong Kong Technology Cooperation Funding Scheme (GHX/075/22GD), by Innovation and Technology Commission (ITC).

Institutional Review Board Statement

Not applicable.

Informed Consent Statement

Not applicable.

Data Availability Statement

The publicly available dataset used in this study can be accessed through Ultralytics at <https://docs.ultralytics.com/datasets/segment/crack-seg>, (accessed on 1 August 2026). The processed data and direction annotations generated during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare no conflict of interest.

Use of AI and AI-Assisted Technologies

During the preparation of this work, the authors used OpenAI Codex to assist with language refinement and L^AT_EX formatting during the proof-correction stage. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

References

1. Yuan, Q.; Shi, Y.; Li, M. A Review of Computer Vision-Based Crack Detection Methods in Civil Infrastructure: Progress and Challenges. *Remote Sens.* **2024**, *16*, 2910. <https://doi.org/10.3390/rs16162910>.
2. Deng, J.; Singh, A.; Zhou, Y.; et al. Review on Computer Vision-Based Crack Detection and Quantification Methodologies for Civil Structures. *Constr. Build. Mater.* **2022**, *356*, 129238. <https://doi.org/10.1016/j.conbuildmat.2022.129238>.
3. Feng, J.; Qi, F.; Zhu, F.; et al. Uncertainty-Informed Cascaded Diagnosis of Compound Faults in Electromechanical Systems via a Physics-Informed Hypergraph Framework. *Reliab. Eng. Syst. Saf.* **2026**, *272*, 112685. <https://doi.org/10.1016/j.res.2026.112685>.
4. Fan, J.; Zheng, P.; Li, S. Vision-Based Holistic Scene Understanding towards Proactive Human–Robot Collaboration. *Robot. Comput.-Integr. Manuf.* **2022**, *75*, 102304. <https://doi.org/10.1016/j.rcim.2021.102304>.
5. Tang, W.; Jahanshahi, M.R. Autonomous Robotic Inspection Based on Active Vision and Deep Reinforcement Learning. In Proceedings of the 14th International Workshop on Structural Health Monitoring, Stanford, CA, USA, 12–14 September 2023. <https://doi.org/10.12783/shm2023/36973>.
6. Wang, Y.; Peng, T.; Wang, W.; et al. High-Efficient View Planning for Surface Inspection Based on Parallel Deep Reinforcement Learning. *Adv. Eng. Inform.* **2023**, *55*, 101849. <https://doi.org/10.1016/j.aei.2022.101849>.
7. Tang, W.; Jahanshahi, M.R. Active Perception Based on Deep Reinforcement Learning for Autonomous Robotic Damage Inspection. *Mach. Vis. Appl.* **2024**, *35*, 110. <https://doi.org/10.1007/s00138-024-01591-7>.
8. Cao, J.; He, H.; Zhang, Y.; et al. Crack Detection in Ultrahigh-Performance Concrete Using Robust Principal Component Analysis and Characteristic Evaluation in the Frequency Domain. *Struct. Health Monit.* **2023**, *23*, 1013–1024. <https://doi.org/10.1177/14759217231178457>.
9. Jin, S.; Lee, S.E.; Hong, J.W. A Vision-Based Approach for Autonomous Crack Width Measurement with Flexible Kernel. *Autom. Constr.* **2020**, *110*, 103019. <https://doi.org/10.1016/j.autcon.2019.103019>.
10. Payab, M.; Abbasina, R.; Khanzadi, M. A Brief Review and a New Graph-Based Image Analysis for Concrete Crack Quantification. *Arch. Comput. Methods Eng.* **2019**, *26*, 347–365. <https://doi.org/10.1007/s11831-018-9263-6>.
11. Tang, W.; Jahanshahi, M.R. Uncertainty-Aware Autonomous Robotic Inspection Based on Active Vision and Deep Reinforcement Learning. In Proceedings of the 15th International Workshop on Structural Health Monitoring, Stanford, CA, USA, 9–11 September 2025. <https://doi.org/10.12783/shm2025/37531>.
12. Li, X.; Gandhi, B.; Zhan, M.; et al. Fine-Tuning Vision-Language Models for Visual Navigation Assistance. *arXiv* **2025**, arXiv:2509.07488.
13. Wang, X.; Dengxiong, X.; Bai, S.; et al. VLabot: A Human Vision–Language–Action Models Interaction Framework for Robotic Assembly. *Robot. Comput.-Integr. Manuf.* **2026**, *100*, 103268. <https://doi.org/10.1016/j.rcim.2026.103268>.
14. Ren, M.; Zheng, P.; Colley, M. VLM-Driven Risk-Adaptive HUD Interactions for Trust Calibration in Automated Driving. In Proceedings of the Extended Abstracts of the 2026 CHI Conference on Human Factors in Computing Systems, Barcelona, Spain, 13–17 April 2026. <https://doi.org/10.1145/3772363.3798997>.
15. Xu, H.; Zhou, L.; Zhou, X.; et al. An Augmented Reality-Enhanced Large Language Model-Based Interactive System for Human-Robot Collaboration. *SSRN* **2026**, 6511489. <https://doi.org/10.2139/ssrn.6511489>.

16. Xia, L.; Hu, Y.; Pang, J.; et al. Leveraging Large Language Models to Empower Bayesian Networks for Reliable Human-Robot Collaborative Disassembly Sequence Planning in Remanufacturing. *IEEE Trans. Ind. Inform.* **2025**, *21*, 3117–3126. <https://doi.org/10.1109/tii.2024.3523551>.
17. Sung, Y.L.; Cho, J.; Bansal, M. VL-Adapter: Parameter-Efficient Transfer Learning for Vision-and-Language Tasks. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 5227–5237. <https://doi.org/10.1109/cvpr52688.2022.00516>.
18. Zanella, M.; Ben Ayed, I. Low-Rank Few-Shot Adaptation of Vision-Language Models. In Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 17–18 June 2024. <https://doi.org/10.1109/cvprw63382.2024.00166>.
19. Wang, Z.; Pang, J. Fast Measuring Pavement Crack Width by Cascading Principal Component Analysis. *arXiv* **2025**, arXiv:2511.02144.
20. Chen, C.; Seo, H.; Jun, C.H.; et al. Pavement Crack Detection and Classification Based on Fusion Feature of LBP and PCA with SVM. *Int. J. Pavement Eng.* **2022**, *23*, 3274–3283. <https://doi.org/10.1080/10298436.2021.1888092>.
21. Weng, X.; Huang, Y.; Wang, W. Segment-Based Pavement Crack Quantification. *Autom. Constr.* **2019**, *105*, 102819. <https://doi.org/10.1016/j.autcon.2019.04.014>.
22. Zeng, X.; Zaenker, T.; Bennewitz, M. Deep Reinforcement Learning for Next-Best-View Planning in Agricultural Applications. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 2323–2329.
23. Qiao, Y.; Yu, Z.; Wu, Q. VLN-PETL: Parameter-Efficient Transfer Learning for Vision-and-Language Navigation. In Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 1–6 October 2023; pp. 15397–15406. <https://doi.org/10.1109/iccv51070.2023.01416>.
24. Hancock, A.J.; Wu, X.; Zha, L.; et al. Actions as Language: Fine-Tuning VLMs into VLAs without Catastrophic Forgetting. *arXiv* **2025**, arXiv:2509.22195.
25. Ghiasvand, S.; Oskouie, H.E.; Alizadeh, M.; et al. Few-Shot Adversarial Low-Rank Fine-Tuning of Vision-Language Models. *arXiv* **2025**, arXiv:2505.15130.
26. Ji, Y.; Liu, Y.; Zhang, Z.; et al. Enhancing Adversarial Robustness of Vision-Language Models through Low-Rank Adaptation. In Proceedings of the International Conference on Multimedia Retrieval, Chicago, IL, USA, 30 June–3 July 2025. <https://doi.org/10.1145/3731715.3733324>.
27. Umrajkar, V. DAC-LoRA: Dynamic Adversarial Curriculum for Efficient and Robust Few-Shot Adaptation. *arXiv* **2025**, arXiv:2509.20792.
28. Fu, J.; Fang, J.; Sun, J. LoFT: LoRA-Based Efficient and Robust Fine-Tuning Framework for Adversarial Training. In Proceedings of the 2024 International Joint Conference on Neural Networks, Yokohama, Japan, 30 June–5 July 2024; pp. 1–8.
29. Yuan, Z.; Zhang, J.; Shan, S. FullLoRA: Efficiently Boosting the Robustness of Pretrained Vision Transformers. *IEEE Trans. Image Process.* **2025**, *34*, 4580–4590. <https://doi.org/10.1109/tip.2025.3587598>.
30. Dunn, E.; Frahm, J.M. Next Best View Planning for Active Model Improvement. In Proceedings of the British Machine Vision Conference, London, UK, 7–10 September 2009; pp. 1–11.
31. Zeng, R.; Wen, Y.; Zhao, W.; et al. View Planning in Robot Active Vision: A Survey of Systems, Algorithms, and Applications. *Comput. Vis. Media* **2020**, *6*, 225–245. <https://doi.org/10.1007/s41095-020-0179-3>.
32. Ultralytics. Crack Segmentation Dataset. Available online: <https://docs.ultralytics.com/datasets/segment/crack-seg> (accessed on 23 August 2026).
33. Yang, A.; Li, A.; Yang, B.; et al. Qwen3 Technical Report. *arXiv* **2025**, arXiv:2505.09388.