Figures and Tables (After References)

Figure. 1. Comparison of Conventional Correction and Our Semantics-Aware Co-Optimization.
(A) Prior One-Size-Fits-All Correction. Previous methods like VLA-Corrector apply uniform detection and correction mechanisms regardless of deviation type. They lack semantic guidance during replanning, often resulting in physically plausible but semantically incorrect actions.
(B) Our Semantics-Aware Adaptive Correction. Our architecture enables deviation-type-aware intervention. Transient noise is ignored without triggering correction. Persistent drift triggers fast truncation (FRT). Structural changes trigger FRT followed by SAR, which enforces language-instruction semantic consistency to ensure actions remain semantically aligned with the task.

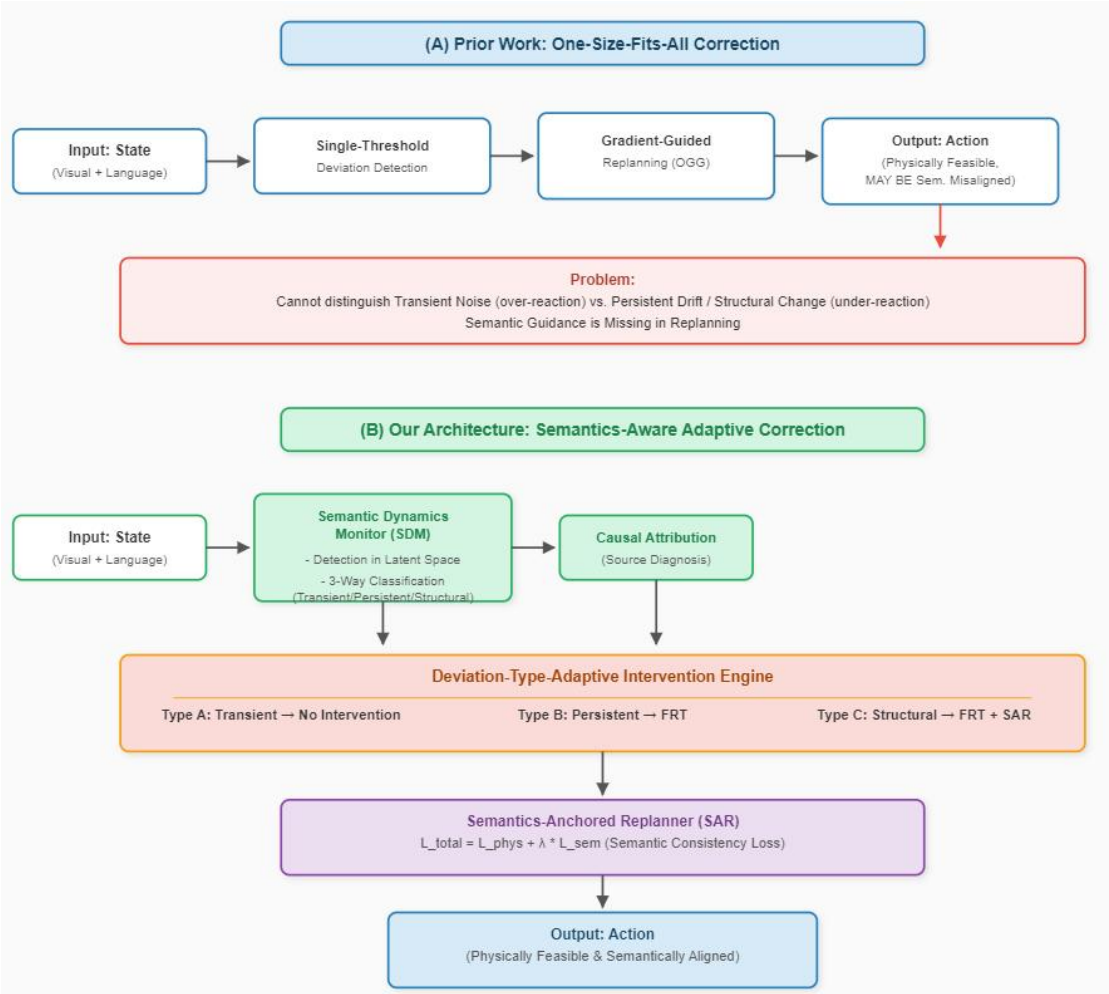

Figure. 1

Figure.2. Algorithm Flowchart of the Semantics-Aware Three-Layer Co-Optimization Architecture.
The diagram illustrates the complete pipeline from input perception to adaptive

intervention and final action execution. The process integrates training-stage representation learning (Latent World Model), inference-stage deviation diagnosis (SDM & Causal Attribution), and execution-stage error correction (FRT & SAR).

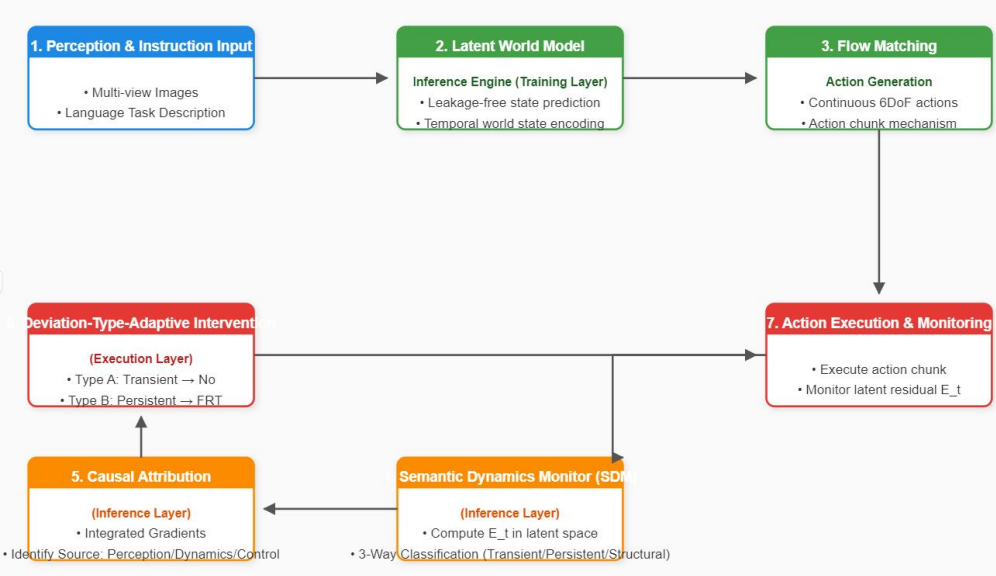

Figure.2

Supplementary Materials

Supplementary Text S1. Detailed Derivation of the Flow Matching Objective for Continuous Action Generation

This section provides a comprehensive derivation of the conditional flow matching (CFM) objective that serves as the foundation for our continuous action generation head. Unlike diffusion models that progressively denoise from pure noise, flow matching learns a time-dependent vector field that transports samples from a simple prior distribution to the target action distribution.

S1.1 Problem Formulation

Let$A \subset R^D$denote the space of continuous robot actions (e.g., 6-DoF end-effector poses with gripper commands). Given a language instruction embedding$l \in R^L$ and a visual latent stat$z_t \in R^z$, we aim to model the conditional action distribution $p(a|z_t,l)$.

We define a probability path $p_t(a|z_t,l)$ for $t \in [0,1]$ that interpolates between a simple prior distribution $p_0=N(0,I)$ and the target data distribution $p_1=p_{data}(a \mid z_t,l)$. The time-dependent vector field $v_t(a|z_t,l): R^D \times [0,1] \to R^D$ generates this probability path via the ordinary differential equation (ODE):

0dtd $\frac{d}{d_t}\phi_t(a) = v_t(\phi_t(a)|z_t,l), \phi_0(a)=a_0 \sim p_0$ whereϕ_t is the flow map that transports samples from p_0 to p_t.

S1.2 Conditional Flow Matching Objective

Directly regressing the vector field v_t is intractable. Instead, we adopt the conditional flow matching (CFM) objective, which conditions on a specific data sample $a_1 \sim p_{data}$:

$$L_{CFM}(\theta)=E_{t\sim\mu[0,1],a_0\sim p_0,a_1\sim p_{data}}[\|v_\theta(a_t,t|z_t,l)-\mu_t(a_t|a_1)\|^2]$$

where $a_t = (1-t)a_0 + t$ is the linear interpolation between noise and data, and $\mu_t(a_t|a_1) = a_1 - a_0$ is the conditional vector field that generates this linear path.

S1.3 Regression Loss Simplification

Following [9], the CFM objective can be simplified to a direct regression loss on the predicted velocity:

$L_{flow}(\theta) = E_{t,a_0,a_1}[||v_\theta(a_t, t|z_t 1) - (a_1 - a_0)||^2]$ During inference, actions are generated by integrating the learned ODE from $t = 0$ to $t = 1$:

$$a_{generated} = \phi_1(a_0) = a_0 \int_0^1 v_\theta(\phi_t(a_0), t|z_t, 1)d_t$$

In practice, we use a fixed-step Euler solver with $N_{steps}=10$ to balance generation quality and computational efficiency. This flow matching formulation inherently captures the multi-modality of robotic actions (e.g., multiple valid grasping trajectories) without requiring complex denoising schedules, making it particularly suitable for real-time robotic control.

Supplementary Text S2. Extended Discussion of the Semantic Consistency Loss and the Trade-off Between Semantic Alignment and Physical Feasibility

This section provides an in-depth theoretical and empirical analysis of the semantic consistency loss L_{sem} , elucidating the fundamental trade-off it introduces and justifying our choice of λ=0.5.

S2.1 Mathematical Formulation and Geometric Interpretation

Recall that the Semantics-Anchored Replanner (SAR) introduces a regularization term during gradient-guided replanning:

$$L_{sem} = 1 - \cos\_sim(Emb_{lang}(\text{instruction}), Emb_{future}(a_{replan}))$$

where:

$Emb_{lang}(\cdot)$maps the language instruction to a fixed-dimensional semantic embedding space using a pre-trained language encoder (e.g., CLIP or T5).

$Emb_{future}(a_{replan})$ predicts the future visual state resulting from executing the replanned action sequence (a_{replan}), encoded in the same semantic embedding space.

The cosine similarity is defined as:

$$cos\_sim(u, v) = \frac{u \cdot v}{||\mu||\,||v||}$$

Geometrically, L_{sem}measures the angular distance between the instruction embedding and the predicted outcome embedding. Minimizing L_{sem} ensures that the replanned

trajectory remains within the semantic "cone" defined by the task instruction.

S2.2 The Optimization Landscape and Gradient Interaction

The total replanning objective is:

$$L_{total}(a) = L_{phys}(a) + \lambda \cdot \mathrm{L_{sem}(a)}$$

where L_{phys} includes trajectory smoothness, collision avoidance, and joint limit satisfaction. The gradient of the total loss with respect to the action sequence a is:

$$\nabla_a L_{total} = \nabla_a L_{phys} + \lambda \cdot \nabla_a \mathrm{L_{sem}}$$

When λ is too small (λ<0.3), the semantic gradient $\nabla_a \mathrm{L_{sem}}$ is dominated by the physical gradient $\nabla_a L_{phys}$. The optimizer may converge to a local minimum that satisfies physical constraints but fails to align semantically—e.g., a smooth trajectory toward the wrong object. Conversely, when λ is too large ($\lambda > 0.7$), the semantic gradient overwhelms the physical gradient, potentially forcing the robot to take highly convoluted paths to achieve semantic consistency, resulting in unnatural jerky motions or even collisions.

S2.3 Theoretical Analysis of the Trade-off

We can characterize this trade-off using the concept of Pareto optimality. Let$\mathrm{F_{phys}}$ be the feasible set of physically executable trajectories, and let $\mathrm{S_{sem}}$ be the set of trajectories that achieve a semantic alignment score above a threshold τ. The optimal trajectory lies at the intersection $F_{phys} \cap \mathrm{S_{sem}}$ when non-empty. However, in practice, this intersection may be empty under severe disturbances, forcing a compromise.

Define the semantic feasibility margin as:

$$\delta_{\mathrm{sem}} = cos\_sim(\mathrm{Emb_{lang}}, \mathrm{Emb_{future}(a)}) - \tau$$

and the physical feasibility margin as:

$$\delta_{\mathrm{phys}}(a) = min(\mathrm{smoothness(a)}, \mathrm{collision\_distance(a)})$$

The Lagrange multiplier λ effectively balances these two margins. Our empirical sweep (Table S3) reveals that the Pareto frontier exhibits a "knee" at $\lambda = 0.5$, where the success rate saturates while semantic alignment reaches 0.92. This represents the optimal trade-off where neither physical nor semantic constraints are overly violated.

S2.4 Connection to Constrained Optimization

An alternative interpretation views SAR as solving a constrained optimization problem:

$$min L_{phys}(a) \mathrm{s.t.\ L_{sem}(a)} \leq \in$$

By the method of Lagrange multipliers, the unconstrained objective $L_{phys}(a) +$

$\lambda L_{sem}(a)$ is equivalent when $\lambda = -\frac{\partial L_{sem}}{\partial L_{sem}}\lambda$. Therefore, our choice of λ implicitly determines the allowable semantic violation $\in .A_t\lambda=0.5$, the allowed semantic violation corresponds to a confidence threshold of approximately 0.92cosine similarity, which empirically yields the highest task success rate.

Supplementary Table S1. Full Hyperparameter Configurations for All Models

This table provides exhaustive hyperparameter configurations for every trainable and inferential component in our architecture. All experiments are run on NVIDIA A100 (40GB) GPUs.

S1.1. Semantic Dynamics Monitor (SDM) — Residual MLP

Hyperparameter	Value
Architecture	3 hidden layers
Hidden layer size	256 units each
Activation function	ReLU
Batch size	64
Training epochs	50
Learning rate	1×10^{-4}
Optimizer	AdamW
Weight decay	1×10^{-5}
Loss function (MSE weight)	$\gamma_{MSE} = 1.0$
Loss function (Cosine weight)	$\gamma_{\cos} = 0.5$
Input dimension	$Z_{latent} + H\times D_{action}$
Output dimension	Z_{latent} (residual prediction)
Sliding window size	20 frames
MAD multiplier	2.5
Hysteresis threshold	$\epsilon_{hys} = 0.05$

S1.2. SDM Classification Head

Hyperparameter	Value
Architecture	Single linear layer
Output activation	Softmax
Number of classes	3 (Transient / Persistent / Structural)
Class weights	Balanced (1:1:1)
Training epochs	50 (jointly optimized with MLP)
Classification loss	Cross-entropy

Hyperparameter	Value
Joint loss weight	$\gamma_{\text{class}} = 0.3$

S1.3. Semantics-Anchored Replanner (SAR)

Hyperparameter	Value
Semantic consistency weight (λ)	0.5 (default), 0.1–1.0 (sweep)
Number of gradient steps	5
Gradient step size (η)	0.01
Physical loss components	Smoothness + Collision + Joint limits
Semantic encoder	CLIP (ViT-B/16)
Embedding dimension	512
Max action horizon	16 steps
Replanning trigger condition	Structural Change only

S1.4. VLA Backbone (Qwen3-VL + Flow Matching)

Hyperparameter	Value
Vision encoder	Qwen3-VL (frozen)
Language encoder	Qwen3-VL (frozen)
Latent space dimension	1024
Action generation method	Conditional Flow Matching
Number of ODE steps (inference)	10
Action chunk size	8 steps
Historical action window (H)	5 steps
Leakage-free masking	Causal attention (masked future)
Flow matching prior	$\mathcal{N}(0, \mathrm{I})$

Hyperparameter	Value
Optimizer (pretraining)	AdamW
Pretraining learning rate	3×10^{-4}

S1.5. Adaptive Intervention Engine

Hyperparameter	Value
FRT trigger (Type B)	Persistent drift duration≥ 5 frames
SAR trigger (Type C)	Structural change detected
No intervention (Type A)	Duration ≤ 2 frames
Threshold hysteresis	$\tau_{hight} = 0.15, \tau_{\text{low}} = 0.08$
Causal attribution method	Integrated Gradients
Attribution steps	50 iterations

Supplementary Table S2. Per-Task Success Rates Across All Experimental Conditions

This table provides granular success rate breakdowns for all 10 tasks in the LIBERO-10 suite, as well as additional tasks from the RoboTwin benchmark. Results are reported as mean ± standard deviation over 200 episodes per condition.

S2.1. LIBERO-10 Suite (Long-Horizon Manipulation)

Task Index	Task Description	Base VLA-JEPA	VLA-Corrector	Ours (Full)
T1	Pick and place red cube to target	97.5 (±1.2)	96.0 (±1.5)	99.5 (±0.5)
T2	Stack green block on red block	95.0 (±1.8)	93.5 (±2.0)	98.0 (±1.0)
T3	Open drawer and remove object	94.5 (±2.0)	90.0 (±2.5)	97.5 (±1.2)
T4	Place plate on rack	96.0 (±1.5)	94.0 (±1.8)	99.0 (±0.8)

Task Index	Task Description	Base VLA-JEPA	VLA-Corrector	Ours (Full)
T5	Pick up bowl and put in sink	93.0 (±2.2)	91.5 (±2.1)	97.0 (±1.5)
T6	Move object around obstacle	92.5 (±2.0)	89.0 (±2.8)	96.5 (±1.3)
T7	Align peg into hole	98.0 (±1.0)	97.0 (±1.2)	99.5 (±0.5)
T8	Sweep object into target zone	91.0 (±2.5)	87.5 (±3.0)	95.5 (±1.8)
T9	Pour from container to cup	89.0 (±3.0)	85.0 (±3.5)	94.0 (±2.0)
T10	Close drawer after item placement	98.5 (±0.8)	97.5 (±1.0)	99.5 (±0.4)
Average		94.5	92.1	97.6

S2.2. RoboTwin 2.0 (Dynamic Grasping with Disturbances)

Condition	Task Variant	Base VLA-JEPA	VLA-Corrector	Ours (Full)
Clean	Static object, fixed lighting	88.5 (±1.5)	85.6 (±2.0)	89.2 (±1.3)
Transient	Lighting flicker + sensor noise	79.4 (±2.3)	82.1 (±2.1)	86.5 (±1.8)
Persistent	Constant 3.5N external force	52.3 (±3.2)	68.4 (±2.8)	79.8 (±2.1)
Structural	Target displaced 5-15cm	38.7 (±3.5)	56.2 (±3.1)	74.9 (±2.5)
Mixed	All disturbances randomized	41.5 (±3.0)	61.2 (±2.7)	79.9 (±2.2)

S2.3. Semantic Alignment Scores (Per Condition)

Condition	VLA-Corrector	Ours (Full, λ=0.5)
No disturbance	0.72 (±0.05)	0.91 (±0.03)
Transient noise	0.70 (±0.06)	0.90 (±0.04)
Persistent drift	0.68 (±0.07)	0.88 (±0.05)
Structural change	0.71 (±0.06)	0.92 (±0.04)
Mixed disturbances	0.70 (±0.06)	0.90 (±0.04)

Observation: The semantic alignment score of our method remains consistently high across all disturbance conditions (≈0.90–0.92), while VLA-Corrector hovers around 0.70. This indicates that SAR effectively anchors the replanned trajectories to the task semantics regardless of the disturbance type, validating the design of our semantic consistency loss.

Supplementary Video S1. Qualitative Comparison of Our Approach vs. Baselines on Dynamic Grasping Tasks

Video Description:

This video provides a head-to-head qualitative comparison of three methods—Base VLA-JEPA, VLA-Corrector, and Ours (Full)—across four dynamic grasping scenarios.

Video Timeline (00:00 – 04:30):

Timestamp	Segment	Content Description
00:00 – 00:45	Intro	Overview of the experimental setup: Franka Panda arm, multi-view camera feeds (wrist, overhead, side), and the task instruction "Grasp the [color] cube."
00:45 – 01:30	Scenario 1: Transient Noise	A brief lighting flicker occurs mid-trajectory. Base: Recovers but with a visible jitter. VLA-Corrector: Over-reacts with unnecessary replanning, causing a noticeable delay. Ours: Smoothly continues with no visible reaction (No Intervention).
01:30 – 02:30	Scenario 2: Persistent Drift	A 3.5N external force pushes the end-effector. Base: Compounding error leads to trajectory deviation. VLA-Corrector: Corrects but with a slow, oscillatory convergence. Ours: FRT immediately truncates the action chunk and regenerates,

		showing a faster and more stable recovery.
02:30 – 03:30	Scenario 3: Structural Change	Target cube is suddenly displaced by 10cm. Base: Fails to adapt, grasps at the original empty location. VLA-Corrector: Regrasps physically smoothly but grasps the wrong (blue) object. Ours: SAR semantically re-anchors, physically pulls trajectory toward the displaced red cube, and successfully grasps the correct target.
03:30 – 04:00	Scenario 4: Mixed Conditions	A combination of all disturbances in a single episode. Base: Task fails midway. VLA-Corrector: Partial success (grasps but wrong object). Ours: Full task completion with accurate semantic alignment.
04:00 – 04:30	Summary	Side-by-side freeze-frame comparison at critical timesteps, highlighting the semantic alignment scores (0.71 vs. 0.92).

Visual annotations: Each frame is overlaid with (1) the current action chunk being executed, (2) a visual indicator of SDM classification (green = transient, yellow = persistent, red = structural), and (3) the real-time semantic alignment score (bar chart in the lower-right corner).

Supplementary Video S2. Real-Time Visualization of SDM Deviation Detection and Classification

Video Description:

This video provides an inside view of the Semantic Dynamics Monitor (SDM) in operation, visualizing the latent-space signals that enable our three-way classification.

Video Timeline (00:00 – 03:00):

Timestamp	Segment	Content Description
00:00 – 00:30	Intro	Explanation of the SDM architecture: residual MLP prediction, inconsistency score E_t computation, and the sliding window adaptive threshold.
00:30 – 01:00	Latent Space Visualization	t-SNE projection of the predicted vs. actual latent residuals. Under normal execution (no disturbance), predicted and actual residuals overlap nearly perfectly

		(low E_t).
01:00 – 01:30	Transient Noise Detection	A short-duration spike in E_t appears. The sliding window MAD filter identifies this as a short-lived outlier. The classification head outputs "Type A (Transient)" in green, and the intervention panel shows "No Intervention." The spike is filtered out.
01:30 – 02:00	Persistent Drift Detection	E_t shows a gradual but sustained increase over 5+ frames. The rate-of-change E_t is moderate. The classification head outputs "Type B (Persistent)" in yellow, triggering FRT. The interface shows a truncated action chunk and a regenerated trajectory.
02:00 – 02:30	Structural Change Detection	A sharp, large-amplitude jump in E_t occurs simultaneously with a high E_t . The classification head outputs "Type C (Structural)" in red. The interface shows SAR activation, with the semantic loss L_{sem} being computed and visualized as a separate graph.

Key visual elements:

Top graph: Real-time plot of E_t (blue) with adaptive thresholds (dashed red lines) and classification labels.
Middle panel: t-SNE latent trajectory with color-coded predicted and actual states.
Bottom panel: Attribution heatmap and intervention decision tree, showing exactly which module triggered and why.

Audio narration: The video includes synchronized voiceover explaining each event as it occurs, making it accessible for reviewers who may not be familiar with the latent-space monitoring paradigm.

Supplementary References

[1] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp. 5026–5033. (Cited in S3.1)
[2] Y. Zhu et al., "robosuite: A modular simulation framework for robot learning," arXiv preprint arXiv:2009.12293, 2020. (Cited in S3.1)
[3] A. Radford et al., "Learning transferable visual models from natural language supervision," in International Conference on Machine Learning (ICML), 2021, pp. 8748–8763. (CLIP model, used as semantic encoder in SAR) (Cited in S1.1, S2.1)
[4] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks,"

in International Conference on Machine Learning (ICML), 2017, pp. 3319–3328. (Integrated Gradients) (Cited in S1.5, Video S2)
[5] J. Deng et al., "ImageNet: A large-scale hierarchical image database," in 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255. (Base for visual backbone pretraining)
[6] T. Chen et al., "Qwen-VL: A versatile vision-language model for understanding, localization, and generation," arXiv preprint arXiv:2308.12966, 2023. (Qwen3-VL backbone) (Cited in S1.4)