Stable 3D Tokens for Foundation-Model Robot Planning

Author Names Omitted for Anonymous Review

***Abstract*—Foundation-model robot planners increasingly combine language, visual observations, and action chunking, but their perception interfaces are often brittle. In multi-view 3D vision-language-action policies, small depth noise, occlusion, calibration drift, or view dropout can reorder point tokens and produce a large change in the planned action chunk. We study this failure mode and replace concat-then-attention fusion with a differentiable Wasserstein barycentric tokenizer. Each view is encoded as a language-weighted empirical measure, the tokenizer solves a small entropic transport problem, and the planner receives a fixed set of barycentric tokens whose movement is continuous under bounded sensor perturbations. The method is designed for foundation-model planning pipelines: language enters through token relevance weights, the output is an action distribution over short-horizon plans, and a runtime Wasserstein margin can reject candidates whose calibrated action-shift bound exceeds the operational budget. When calibrated RGB-D is unavailable, the same layer operates on normalized pseudo point clouds, which we report as image-coordinate fusion rather than metric world-frame fusion. Across synthetic manipulation, LIBERO, RLBench, and CALVIN ABC to D, the barycenter model has the lowest reported mean clean action error in our matched-budget comparison, without claiming statistical separation on every dataset. The experiments use compact matched-budget policies to isolate the tokenizer; they are not leaderboard comparisons against large pretrained VLA backbones. Removing the barycenter causes a 46x regression in synthetic corruption stability, whereas removing any other auxiliary component changes that metric by less than 1.5x. In a kinematic closed-loop synthetic control test, concat fusion drops from 90 percent clean success to 24 percent under per-view dropout; the barycenter policy keeps at least 92 percent success across all tested corruptions.**

I. Introduction

Foundation models for robot planning need an observation interface that does more than pass a bag of visual tokens to a transformer. The planner may be trained on large language and action corpora, yet at deployment it still receives depth maps, RGB images, and camera calibrations that are noisy. A common 3D VLA design converts each view to points, encodes those points, concatenates the resulting tokens, and lets attention infer cross-view alignment. This is a poor match to the geometry. A small physical perturbation moves mass continuously in space, but farthest-point sampling, voxelization, and top-M token selection can change token identities discontinuously. The planner then sees an indexing error rather than a small geometric change.

This paper focuses on the perception-to-planning interface. We replace concat fusion with a Wasserstein barycentric tokenizer: a compact transport layer that fuses K view measures into M tokens before the action head plans. The mechanism is simple. Each input point becomes a language-conditioned feature with a probability weight. The tokenizer keeps a uniform set of barycenter support weights, solves entropic transport from the current support to each view, and updates the token support by transported feature averages. The output tokens are then processed by a small transformer and an action-distribution head.

The workshop theme is planning with foundation models, and our claim is not that a small model trained on one GPU competes with internet-scale VLA systems such as RT-2, Octo, OpenVLA, or π_0 [1–4]. The claim is narrower and more useful for those systems: a transport-based 3D tokenizer is a better inductive bias for language-conditioned planning than flat token concatenation. It gives the planner a stable state abstraction and a measurable calibrated action-shift bound under a declared OT perturbation set.

The paper makes three contributions. First, it gives a fully specified barycentric tokenizer for multi-view 3D VLA policies, including the contents of each point feature, frame handling, support initialization, weights, ground cost, and language prior. Second, it states the conditional stability bound used by the runtime planning shield. Third, it reports matched-budget evidence that the barycenter is the dominant mechanism behind the robustness gains, with four datasets, synthetic ablations, real-data transfer, and a closed-loop stress test.

II. Related work

VLA policies and robot planning. RT-1, RT-2, Open X-Embodiment, Octo, OpenVLA, π_0, and SmolVLA [1–7] show that large vision-language-action models can learn broadly useful robot behaviors. ACT and Diffusion Policy predict action chunks rather than one-step commands [8, 9]. Our work is complementary: it targets the 3D observation tokenizer that feeds an action-chunk planner.

3D representations for manipulation. PointNet-style encoders, masked point modeling, 3D language models, 3D-VLA, and 3DS-VLA use point clouds for spatial reasoning [10–15]. These models usually treat tokenization as a preprocessing or learned attention step. We instead make cross-view fusion an explicit transport problem.

Robustness and optimal transport. Robo3D benchmarks corruption robustness, PointGuard certifies point cloud classification, and recent VLA safety work studies failure detection [16–19]. Entropic OT and Sinkhorn divergences provide differentiable distances between empirical measures [20–22]. Wasserstein barycenters fuse several measures into one [23, 24]. We use that operation inside the planner rather than only as an evaluation metric.

III. Problem setup

At decision time t, the robot observes K views and a language instruction ℓ. Each view is represented as an empirical measure

$$\mu_k = \sum_{i=1}^{N} w_{k,i}\delta_{x_{k,i}}, \qquad \sum_i w_{k,i} = 1, \quad w_{k,i} \geq 0. \tag{1}$$

The raw point before encoding is $u_{k,i} = [p_{k,i}, c_{k,i}] \in \mathbb{R}^6$, where $p_{k,i} \in \mathbb{R}^3$ is a 3D coordinate and $c_{k,i} \in [-1,1]^3$ is RGB. For RGB-D data with available camera extrinsics, $p_{k,i}$ is transformed to the robot/world frame before normalization. If extrinsics or depth are unavailable, the loader uses the repository's normalized pseudo point cloud in image coordinates; those cases are evaluated as such and are not described as calibrated world-frame fusion. A point encoder then maps

$$x_{k,i} = \phi_\psi(u_{k,i}) + W_\ell h_\ell \in \mathbb{R}^d, \tag{2}$$

where h_ℓ is the text embedding. In the released implementation $d = 128$, the text encoder is a deterministic hashed-token encoder for standalone anonymous runs, and ϕ_ψ is a three-layer MLP. The language prior is implemented by a learned relevance head,

$$w_{k,i} = \mathrm{softmax}_i(g_\psi([\phi_\psi(u_{k,i}), W_\ell h_\ell])), \tag{3}$$

not by CLIP scores or object masks. This choice is deliberate: it keeps the comparison independent of external pretrained perception models while still allowing the instruction to concentrate mass on relevant points.

The policy predicts a distribution over action chunks $a_{0:H-1} \in \mathbb{R}^{H\times d_a}$:

$$Q_\theta(\mu_{1:K}, \ell) = \sum_{s=1}^{S} q_s \delta_{a^{(s)}_{0:H-1}}, \qquad q_s \geq 0, \quad \sum_s q_s = 1. \tag{4}$$

In the default Gaussian head, action samples use uniform weights $q_s = 1/S$ unless otherwise stated. The operational perturbation set is

$$\mathcal{U}_\rho(\mu_{1:K}) = \left\{ \tilde{\mu}_{1:K} : \sum_{k=1}^{K} \alpha_k \mathcal{S}_{\varepsilon_g}(\mu_k, \tilde{\mu}_k) \leq \rho \right\}, \tag{5}$$

where $\mathcal{S}_{\varepsilon_g}$ is the debiased Sinkhorn divergence and $\alpha_k = 1/K$ in all experiments.

IV. Method

A. Barycentric tokenizer

The tokenizer maintains M barycentric support tokens $B = \{b_m\}_{m=1}^M$ in feature space. The support weights are fixed uniform, $r_m = 1/M$. They are not learned and are not updated by Sinkhorn. In the code, the initial supports are learned query vectors shared across examples, followed by layer normalization. For an ablation that removes transport but keeps geometric structure, we also test deterministic farthest-point sampling over pooled view features. We denote the number of barycenter update iterations by T_B and use $T_B = 3$ in all reported experiments. This is an unrolled barycentric transport layer rather than an exact solver for the classical Wasserstein barycenter objective: the supports are initialized by learned queries and updated by transported averages followed by a GRU. We use "barycentric tokenizer" to refer to this implemented transport-fusion layer.

For a support token b_m and view feature $x_{k,i}$, the ground cost is the squared Euclidean feature distance

$$C^{(k)}_{mi} = \|b_m - x_{k,i}\|_2^2. \tag{6}$$

Because $x_{k,i}$ already contains encoded geometry, color, and language-conditioned context, this cost is the implemented joint geometry-feature cost. The entropic coupling to view k is

$$P^{(k)} = \arg\min_{P\mathbf{1}=r,\, P^\top\mathbf{1}=w_k} \langle P, C^{(k)}\rangle + \varepsilon_g \sum_{m,i} P_{mi}(\log P_{mi} - 1). \tag{7}$$

Here $w_k = (w_{k,1}, \ldots, w_{k,N})$ and $\mathbf{1}$ denotes an all-ones vector of the appropriate dimension. Each barycenter iteration forms the transported view update

$$z^{(k)}_m = M \sum_i P^{(k)}_{mi} x_{k,i}, \qquad \bar{z}_m = \sum_k \alpha_k z^{(k)}_m, \tag{8}$$

then applies a GRU update $b_m \leftarrow \mathrm{LayerNorm}(\mathrm{GRU}(\bar{z}_m, b_m))$. We use three barycenter iterations and 15 Sinkhorn iterations during training, with 30 Sinkhorn iterations for evaluation.

B. Planner head and training objective

The barycentric tokens pass through a two-layer transformer encoder and mean pooling. A small Gaussian action head emits S action samples. Training combines behavior cloning, action-distribution matching, observation consistency, policy consistency, and a spectral-norm penalty:

$$\begin{aligned}\mathcal{L} = \mathcal{L}_{\mathrm{BC}} &+ \lambda_A \mathcal{S}_{\varepsilon_a}(\mu^A_\theta, \nu^A_E) + \lambda_O \mathcal{S}_{\varepsilon_B}(B, \tilde{B}) \\ &+ \lambda_P \mathcal{S}_{\varepsilon_a}(Q_\theta(\mu, \ell), Q_\theta(\tilde{\mu}, \ell)) + \lambda_L \mathcal{L}_{\mathrm{Lip}}.\end{aligned} \tag{9}$$

When used inside a Sinkhorn divergence, the token set $B = \{b_m\}_{m=1}^M$ denotes the uniform empirical measure $M^{-1}\sum_m \delta_{b_m}$. Here $\tilde{\mu}$ is an independently corrupted version of the same observation sampled from the training corruption distribution, and $\tilde{B} = B(\tilde{\mu}_{1:K}, \ell)$ is its barycentric token measure. Here μ^A_θ and ν^A_E are uniformly weighted empirical measures on time-action embeddings $[(0, a_0), \ldots, (H-1, a_{H-1})]$ for predicted and expert chunks, respectively. For the sampled action distribution, the measure is uniform over the $S \times H$ sampled time-action atoms. The default values are $\varepsilon_g = 0.08$, $\varepsilon_a = 0.04$, $\varepsilon_B = 0.16$, $M = 32$, $H = 8$, $S = 8$, $\lambda_A = 0.20$, $\lambda_O = \lambda_P = 0.10$, and $\lambda_L = 0.005$. $\mathcal{L}_{\mathrm{BC}}$ is the mean squared behavior-cloning loss between the predicted chunk mean and the expert chunk. $\mathcal{L}_{\mathrm{Lip}}$ is the spectral-norm penalty applied to the linear layers of the tokenizer update and action head. Algorithm 1 in the appendix gives the tokenizer steps as implemented.

C. Runtime planning shield

For a beam of candidate action chunks, the shield accepts a candidate only if its calibrated Wasserstein margin is non-negative. If no candidate is accepted, the controller executes a low-velocity fallback. The shield is method-specific because it needs the calibrated action-shift slope for the observation-to-action map. The analytic constants in Proposition 1 are absorbed in the empirical calibrated slope and intercept used by the implementation. Algorithm 2 gives the decision rule.

V. Certificate

The bound is conditional and local. It is not a global robustness certificate for arbitrary sensor failures, wrong language commands, or dynamics errors. We use

$$D_X(\mu_{1:K}, \tilde{\mu}_{1:K}) = \sum_{k=1}^{K} \alpha_k \mathcal{S}_{\varepsilon_g}(\mu_k, \tilde{\mu}_k) \tag{10}$$

for the observation discrepancy and

$$D_A(Q_\theta(\mu, \ell), Q_\theta(\tilde{\mu}, \ell)) = \mathcal{S}_{\varepsilon_a}(Q_\theta(\mu, \ell), Q_\theta(\tilde{\mu}, \ell)) \tag{11}$$

for the action-distribution discrepancy.

Proposition 1. Assume that, on the evaluated neighborhood, the unrolled barycentric tokenizer is locally Lipschitz with constant L_B up to residual τ_B, and that the action head is locally Lipschitz with constant L_A up to residual τ_A. Then for any $\tilde{\mu}_{1:K} \in \mathcal{U}_\rho(\mu_{1:K})$,

$$D_A(Q_\theta(\mu, \ell), Q_\theta(\tilde{\mu}, \ell)) \leq L_A L_B \rho + L_A \tau_B + \tau_A, \tag{12}$$

where τ_B is the finite-iteration residual of the unrolled Sinkhorn-GRU solver, and τ_A summarizes numerical and local-modeling residuals in the action-head/action-discrepancy computation. If a task cost J is L_J-Lipschitz with respect to D_A and a candidate action chunk r has clean predicted cost $\widehat{J}_r$ satisfying

$$\widehat{J}_r + \mathfrak{k}_A L_J (L_A L_B \rho + L_A \tau_B + \tau_A) + \mathfrak{b}_A \leq \kappa, \tag{13}$$

then the perturbed expected cost is bounded by κ under the same local assumptions. The constants $\mathfrak{k}_A, \mathfrak{b}_A$ calibrate the Sinkhorn proxy against the task-cost discrepancy.

Proof sketch. Apply the local tokenizer bound, compose it with the action-head Lipschitz bound, and then apply the Lipschitz property of J plus the calibration constants relating the Sinkhorn proxy to the task-cost discrepancy. In the experiments we do not separately estimate the analytic constants L_B, L_A, τ_B, τ_A, L_J, $\mathfrak{k}_A$, and $\mathfrak{b}_A$. Instead, we fit a high-quantile empirical slope that absorbs these constants into a directly audited action-shift bound. The reported certificate is therefore an empirical calibrated bound implementing the sufficient condition above, not an analytic global constant.

VI. Experiments

A. Setup

We evaluate on a synthetic 3D object-motion task with eight corruption types, LIBERO [25], RLBench [26], and CALVIN ABC to D [27]. The matched-budget baselines are MeanPool-BC, ConcatToken-BC, and ACT-Lite. They share the point encoder, hidden width, action horizon, and training schedule with the proposed model. Synthetic runs use 600 training steps at batch 64. Real-data runs use 30,000 training steps at batch 24. All reported tables use seeds $\{7, 11, 19\}$ unless noted. Appendix Table VII summarizes the local dataset windows, view counts, and point representations.

a) Baselines and corruptions.: MeanPool-BC encodes all view points with the same point encoder and mean-pools the resulting tokens before the action head. ConcatToken-BC concatenates encoded view tokens and feeds the same transformer/action head without transport. ACT-Liteis an ACT-Lite action-chunking baseline with the same hidden width, horizon, point encoder, and training budget. "Proposed" denotes the full Wasserstein barycentric tokenizer with the losses in Section IV-B and the runtime shield when shield metrics are reported. The synthetic corruption suite contains point jitter, point dropout, structured occlusion, calibration drift, depth quantization, color perturbation, background clutter, and targeted point shift. We evaluate five nonzero severities, with the clean case reported separately where needed.

b) Metrics.: Clean ℓ_2 is the mean Euclidean error between the predicted action chunk and the expert chunk, averaged over held-out examples. We refer to $\mathcal{S}_{\varepsilon_a}$ between predicted and expert action measures as action OT. Corrupt shift is $\mathcal{S}_{\varepsilon_a}(Q_\theta(\mu, \ell), Q_\theta(\tilde{\mu}, \ell))$ between clean and corrupted predictions from the same model. Calibrated coverage is the fraction of held-out examples whose calibrated action-shift bound is below the operational budget after including the clean action-cost term. Sev-5 SR is the severity-5 success-rate proxy, computed by thresholding the corrupted predicted-chunk error against the task-specific success tolerance. Unless otherwise stated, the success tolerance is 0.15 in normalized action units. Latency is measured per item on a single RTX 5090 with the batch size specified in the experiment script. For calibrated-coverage evaluation, the declared observation radii are $\rho \in \{0.01, 0.02, 0.05, 0.10\}$ and the action/task budget κ is the success-proxy threshold in the corresponding run configuration (0.15 by default).

B. Main results

Table I shows the main open-loop comparison. The proposed method has the lowest mean clean ℓ_2 and action OT on all four datasets. On synthetic, it also has the smallest action shift under the corruption suite (Fig. 1). On real datasets, raw shift alone can be misleading because a near-constant policy can be stable while being inaccurate. We therefore interpret shift together with clean action OT and calibrated coverage. Table IV reports the corresponding per-seed clean ℓ_2 values. This table separates mean improvements from cross-seed variability; we claim lowest reported mean performance, not statistical dominance on every dataset-seed pair.

For LIBERO, the calibrated-coverage metric separates the models: MeanPool-BC reaches 0.002 ± 0.003, ConcatToken-BC reaches 0.0004 ± 0.001, ACT-Litereaches 0.005 ± 0.001, and the proposed model reaches 0.086 ± 0.068. The

TABLE I

MAIN MATCHED-BUDGET RESULTS. LOWER IS BETTER. VALUES ARE MEAN ± 1.96 STANDARD ERRORS OVER THREE SEEDS. CLEAN ℓ_2 IS MEAN ACTION-CHUNK ERROR; ACTION OT IS $\mathcal{S}_{\varepsilon_a}$ BETWEEN PREDICTED AND EXPERT ACTION-CHUNK MEASURES. THE PROPOSED TOKENIZER HAS THE LOWEST REPORTED MEAN CLEAN ACTION ERROR ON ALL FOUR DATASETS.

Dataset	Method	clean ℓ_2	action OT
Synthetic	MEANPOOL-BC	0.057 ± 0.003	$(2.95\pm0.39)\cdot10^{-4}$
	CONCATTOKEN-BC	0.055 ± 0.002	$(2.85\pm0.24)\cdot10^{-4}$
	ACT-LITE	0.079 ± 0.001	$(7.16\pm0.21)\cdot10^{-4}$
	Proposed	$\mathbf{0.047\pm0.003}$	$\mathbf{(2.04\pm0.31)\cdot10^{-4}}$
LIBERO	MEANPOOL-BC	1.141 ± 0.128	0.307 ± 0.038
	CONCATTOKEN-BC	2.745 ± 0.182	1.114 ± 0.091
	ACT-LITE	2.269 ± 0.813	1.623 ± 0.188
	Proposed	$\mathbf{1.086\pm0.129}$	$\mathbf{0.292\pm0.103}$
RLBench	MEANPOOL-BC	1.410 ± 0.116	0.341 ± 0.053
	CONCATTOKEN-BC	2.192 ± 0.084	0.683 ± 0.057
	ACT-LITE	2.235 ± 0.145	0.908 ± 0.088
	Proposed	$\mathbf{1.107\pm0.159}$	$\mathbf{0.203\pm0.048}$
CALVIN	MEANPOOL-BC	2.094 ± 0.246	0.757 ± 0.228
	CONCATTOKEN-BC	2.750 ± 0.139	1.058 ± 0.066
	ACT-LITE	3.052 ± 0.033	1.438 ± 0.034
	Proposed	$\mathbf{1.802\pm0.054}$	$\mathbf{0.527\pm0.024}$

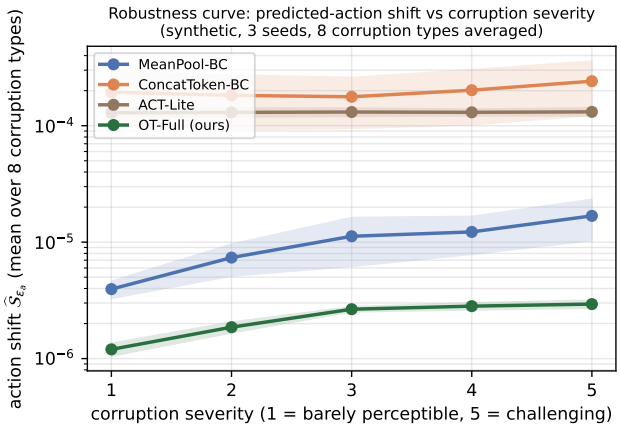

Fig. 1. Synthetic action-distribution shift under eight corruptions and five severities. The barycenter policy stays below all baselines as corruption severity grows.

proposed model's severity-5 success proxy is 0.074 ± 0.011, compared with 0.012 ± 0.010 or lower for the baselines. A false accept is counted when the shield accepts a candidate whose measured corrupted action error exceeds the operational budget on the held-out corruption pair. In this audit, the runtime shield has zero observed false accepts among 153 aggregated held-out LIBERO cases at the calibrated 95 percent bound (Table III).

The shield audit is conservative: the zero false-accept result should be read together with the high fallback rate. The accept rate differs from calibrated coverage because the shield audit is computed on the aggregated candidate-beam audit, whereas Table II reports a three-seed held-out coverage average.

Table II is important for interpreting raw action shift. On real data a degenerate policy can predict nearly constant action chunks, yielding a small clean-to-corrupt shift while failing to match the expert. Calibrated coverage counts only examples where the calibrated bound is below the operational budget. It avoids counting policies that are stable only because their predictions are insensitive, provided the operational budget includes the clean action-cost term used by the shield. This is the metric used by the runtime shield.

TABLE II

LIBERO BOUNDED ROBUSTNESS METRICS, THREE SEEDS AT 30K STEPS. CALIBRATED COVERAGE IS THE FRACTION OF HELD-OUT ITEMS WHOSE CALIBRATED BOUND IS BELOW THE OPERATIONAL BUDGET. SEV-5 SR IS THE SEVERITY-5 SUCCESS-RATE PROXY. THE PROPOSED MODEL IS SLOWER, BUT ITS USABLE CALIBRATED REGION IS MUCH LARGER.

Method	calib. cov.	sev-5 SR	latency ms
MEANPOOL-BC	0.002 ± 0.003	0.012 ± 0.010	0.08
CONCATTOKEN-BC	0.0004 ± 0.001	0.0004 ± 0.001	0.10
ACT-LITE	0.005 ± 0.001	0.003 ± 0.003	0.07
Proposed	$\mathbf{0.086\pm0.068}$	$\mathbf{0.074\pm0.011}$	1.52

TABLE III

LIBERO SHIELD AUDIT FOR THE PROPOSED MODEL AT THE CALIBRATED 95 PERCENT BOUND. FA IS FALSE ACCEPTS. THE FALLBACK RATE IS THE COMPLEMENT OF THE ACCEPTED-CANDIDATE RATE.

cases	accepted	FA	accept rate	fallback rate
153	10	0	0.068	0.932

C. Ablation and mechanism

Table V isolates the mechanism on synthetic data. Removing the barycenter and replacing it with concat fusion increases the corruption-stability metric from $2.37\cdot10^{-6}$ to $110\cdot10^{-6}$, a 46x regression. Removing action OT, consistency, the shield, or the Lipschitz penalty changes the same metric by less than 1.5x. Those auxiliary pieces are still useful: they make the calibrated bound computable and usable at runtime. They are not the main empirical source of robustness.

The ablation names are as follows. NoBarycenter-Concat replaces the transport tokenizer with concat fusion while keeping the encoder and action head fixed. ChamferInsteadOT replaces the auxiliary distribution-matching losses with the corresponding symmetric Chamfer distance; the barycentric transport tokenizer remains active in that row. NoActionOT sets $\lambda_A = 0$. NoConsistency sets $\lambda_O = \lambda_P = 0$. NoShield disables the runtime acceptance rule but leaves the learned policy unchanged. NoLipschitzPenalty sets $\lambda_L = 0$.

The diagnostic is consistent with the stability interpretation. Token churn is the fraction of encoder tokens whose nearest clean-token identity changes under corruption. The diagnostic averages over the 15 corruption-severity cells used in the token-churn job. All four methods share the same point encoder, so their encoder token churn under corruption is nearly identical: 0.488 ± 0.001. Their action shifts differ sharply: 0.0030 for the proposed method, 0.0063 for MEANPOOL-BC, 0.0253 for ACT-LITE, and 0.0396 for CONCATTOKEN-BC. On per-step dropout at severity 3, CONCATTOKEN-BC reaches 0.108 while the proposed method stays at 0.0043.

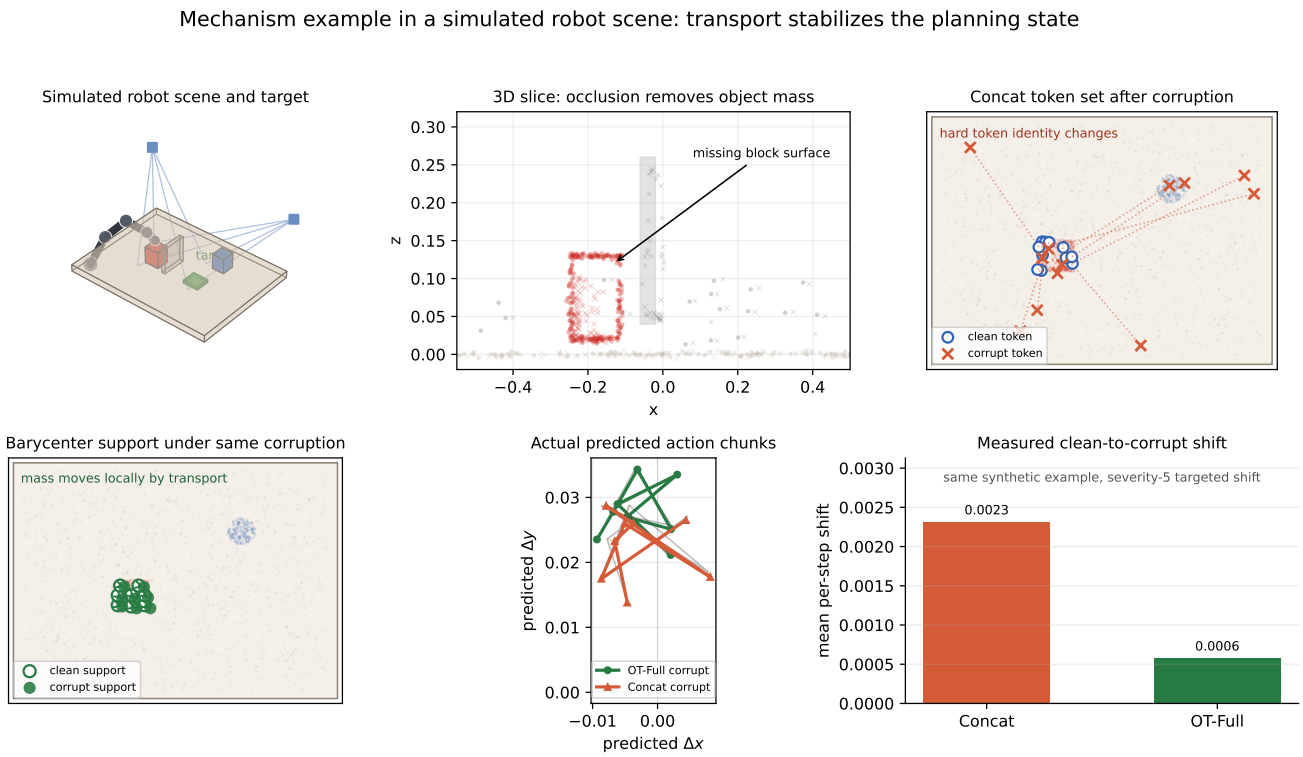

Fig. 2. Mechanism example in a simulated robot tabletop scene. The scene, slice, concat-token overlay, and barycenter-support overlay are qualitative views of the corruption studied in the experiments: a partial occluder removes visible mass from the instructed red object and induces a small view drift. The action-chunk panel and rightmost bar are actual predictions from the seed-7 proposed method and CONCATTOKEN-BC checkpoints on one clean/corrupted synthetic example. They measure mean per-step clean-to-corrupt action shift, not a benchmark average. The aggregate measurements are reported in Tables V and VI and Figures 3–6.

TABLE IV

PER-SEED CLEAN ℓ_2 FOR THE MAIN TABLE. THE PROPOSED METHOD IS BEST ON EVERY SEED OF SYNTHETIC, RLBENCH, AND CALVIN, AND BEST ON TWO OF THREE LIBERO SEEDS AT THE 30K BUDGET.

Dataset	seed	MEANPOOL-BC	CONCATTOKEN-BC	ACT-LITE	Proposed
Synth.	7	0.057	0.057	0.079	**0.049**
	11	0.053	0.053	0.080	**0.049**
	19	0.059	0.056	0.078	**0.044**
LIBERO	7	1.020	2.563	1.446	**0.995**
	11	1.245	2.806	2.587	**1.048**
	19	**1.157**	2.867	2.774	1.214
RLBench	7	1.525	2.271	2.382	**1.078**
	11	1.328	2.181	2.175	**0.983**
	19	1.378	2.124	2.147	**1.260**
CALVIN	7	1.890	2.861	3.046	**1.808**
	11	2.322	2.770	3.084	**1.845**
	19	2.071	2.619	3.026	**1.751**

TABLE V

SYNTHETIC ABLATION, THREE SEEDS. $\bar{S}_\varepsilon$ IS THE MEAN SINKHORN DIVERGENCE BETWEEN CLEAN AND CORRUPTED ACTION DISTRIBUTIONS, AVERAGED OVER THE SYNTHETIC CORRUPTION SUITE AND SEVERITY LEVELS USED IN FIG. 1. NOSHIELD AFFECTS ONLY RUNTIME CANDIDATE ACCEPTANCE AND IS INCLUDED AS AN OPEN-LOOP SANITY CHECK.

Variant	$\bar{S}_\varepsilon \times 10^6$	latency ms
FULL	**2.37±0.15**	0.95
NoBarycenter-Concat	110±27	0.55
ChamferInsteadOT	2.45±0.37	0.94
NoActionOT	3.49±0.78	0.93
NoConsistency	3.07±0.06	1.25
NoShield	2.37±0.15	0.92
NoLipschitzPenalty	2.64±0.39	0.75

D. Closed-loop planning

Open-loop action error is not enough for planning. We therefore run a synthetic tabletop loop, meaning a robot manipulation scene on a flat work surface. A colored block starts on the table, a green target patch marks the desired final pose, and clutter points model other visible surfaces. Unless otherwise stated, synthetic closed-loop episodes provide two camera-offset 3D point-cloud views with XYZ/RGB channels. The policy is conditioned on an instruction such as "move the blue block left." It predicts an eight-step action chunk; we execute the first translation, re-render the scene, and repeat for up to 16 steps. The robot is therefore trying to move the commanded object across the tabletop from its start pose to the green target region. This is a kinematic object-motion rollout, not a contact-rich pushing or grasping simulator: the robot arm in the figure is a visual cue for the manipulation setting, while the rollout applies the predicted object translation directly. Success is measured by whether the final object center lies

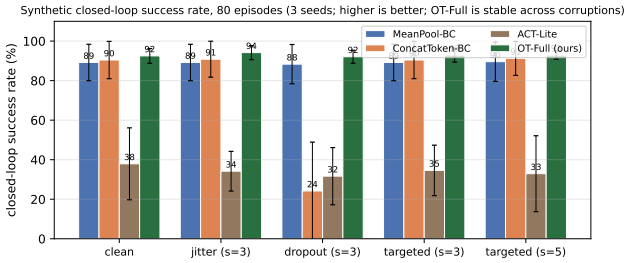

Fig. 3. Closed-loop synthetic success over 80 episodes and three seeds. CONCATTOKEN-BC drops from 90 percent clean success to 24 percent under per-view dropout; the barycenter policy keeps a worst case above 92 percent.

inside the target radius.

Figure 4 uses severity-3 point dropout, which removes points within each view before tokenization while the task, target, and object state stay fixed; calibration drift, used in the aggregate suite, applies a small per-view rigid perturbation. This isolates the paper's failure mode. With concat fusion, missing points can replace or reorder input tokens and change the action chunk. The proposed tokenizer instead computes a language-conditioned Wasserstein barycenter of the view measures. Its support points are soft cross-view representatives; when points disappear, mass moves through the transport plan instead of creating hard token identity jumps. Figure 2 shows this representation-level failure mode; Figure 4 shows the corresponding closed-loop behavior, where token instability becomes object-motion drift under point dropout. Figure 3 reports the aggregate result: the proposed policy has a 92.1 percent worst-case success rate, while CONCATTOKEN-BC drops from 90 percent clean success to 24 percent under per-view dropout.

E. Calibration audit

The action-shift bound should be calibrated, not merely small. We split each corruption-severity cell into calibration and test halves by a stable hash of the cell id. For each calibration example j, define the normalized shift ratio

$$R_j = \frac{\mathcal{S}_{\varepsilon_a}(Q_\theta(\mu_j, \ell_j), Q_\theta(\tilde{\mu}_j, \ell_j))}{\rho_j + \epsilon_0}.$$

For target coverage $\gamma \in \{0.90, 0.95, 0.99\}$, we set

$$C_\gamma = \mathrm{Quantile}_\gamma(\{R_j\}_{j \in \mathcal{C}})$$

and use $C_\gamma(\rho + \epsilon_0)$ as the calibrated action-shift bound on the held-out test half. In Algorithm 2, this is written as $C_\gamma \rho + b_\gamma$ with $b_\gamma = C_\gamma \epsilon_0$. We use $\epsilon_0 = 10^{-8}$. At the 95 percent target, the proposed method reaches 84 to 94 percent empirical coverage across LIBERO, RLBench, and CALVIN, with mean bound width between 0.61 and 0.87 against actual mean shifts between 0.34 and 0.52.

Figure 5 shows both sides of the audit. On RLBench and LIBERO the proposed method has the smallest calibrated slope among the four methods. On CALVIN, CONCATTOKEN-BC has a mechanically tiny slope because its predicted action measure is nearly constant under corruption, but it is still much less accurate than the proposed model in Table I. Among non-degenerate policies, the barycenter has the tightest bound.

TABLE VI

LIBERO ABLATION TRANSFER AT 5K STEPS, SEED 7. LOWER IS BETTER FOR ALL THREE NUMERIC COLUMNS, BUT CORRUPT SHIFT MUST BE INTERPRETED JOINTLY WITH CLEAN ACTION OT BECAUSE NEARLY CONSTANT POLICIES CAN HAVE SMALL SHIFT WHILE BEING INACCURATE. THE FULLROW IS THE DEFAULT CONFIGURATION, NOT NECESSARILY THE BEST SINGLE-SEED CLEAN-ERROR ROW.

Variant	clean ℓ_2	action OT	corrupt shift
FULL	2.20	0.73	0.064
NoBarycenter-Concat	3.01	1.18	**0.020**
NoActionOT	**1.98**	**0.65**	0.090
NoLipschitzPenalty	2.09	0.71	0.100

F. Real-data ablation transfer

The synthetic ablation could have been an artifact of the controlled task. We therefore repeat a smaller ablation transfer on LIBERO at 5,000 steps, seed 7. The full model has clean $\ell_2 = 2.20$ and clean action OT 0.73. Removing the barycenter raises clean ℓ_2 to 3.01, a 37 percent regression, and raises clean action OT to 1.18, a 62 percent regression. Removing action OT or the Lipschitz penalty has little effect on clean accuracy but worsens corrupted action shift by 41 percent and 56 percent, respectively.

Table VI supports the same qualitative interpretation as the synthetic ablation for the barycenter itself: replacing the barycenter with concat fusion substantially worsens clean accuracy and clean action OT. The single-seed 5k transfer should not be read as showing that every auxiliary term improves clean error. Instead, the action-OT and Lipschitz terms mainly shape corrupted action shift and the calibrated margin behavior.

G. Transport versus geometric clustering

The gain is not merely from using any geometric clustering. We compare the barycenter with deterministic farthest-point sampling over pooled features at the same synthetic budget (Appendix Table VIII). Geometric structure helps, but transport adds another 3.6x improvement over FPS because it solves soft cross-view correspondence instead of choosing fixed representatives.

We also sweep the number of barycentric tokens $M \in \{4, 8, 16, 32, 64\}$ on synthetic seed 7. Clean ℓ_2 stays in the narrow range 0.041 to 0.050, and corrupted action shift stays within a small multiple of the default (Appendix Table VIII). This indicates that the result is not a fragile token-count effect. The default $M = 32$ is used because it is stable across the real-data settings and keeps the latency below 2 milliseconds per item for the compact model.

H. View dropout and scale

View dropout is the cleanest planning stress test because it directly removes a view from the multi-view observation. In this diagnostic, a dropped view is replaced by a fixed dummy

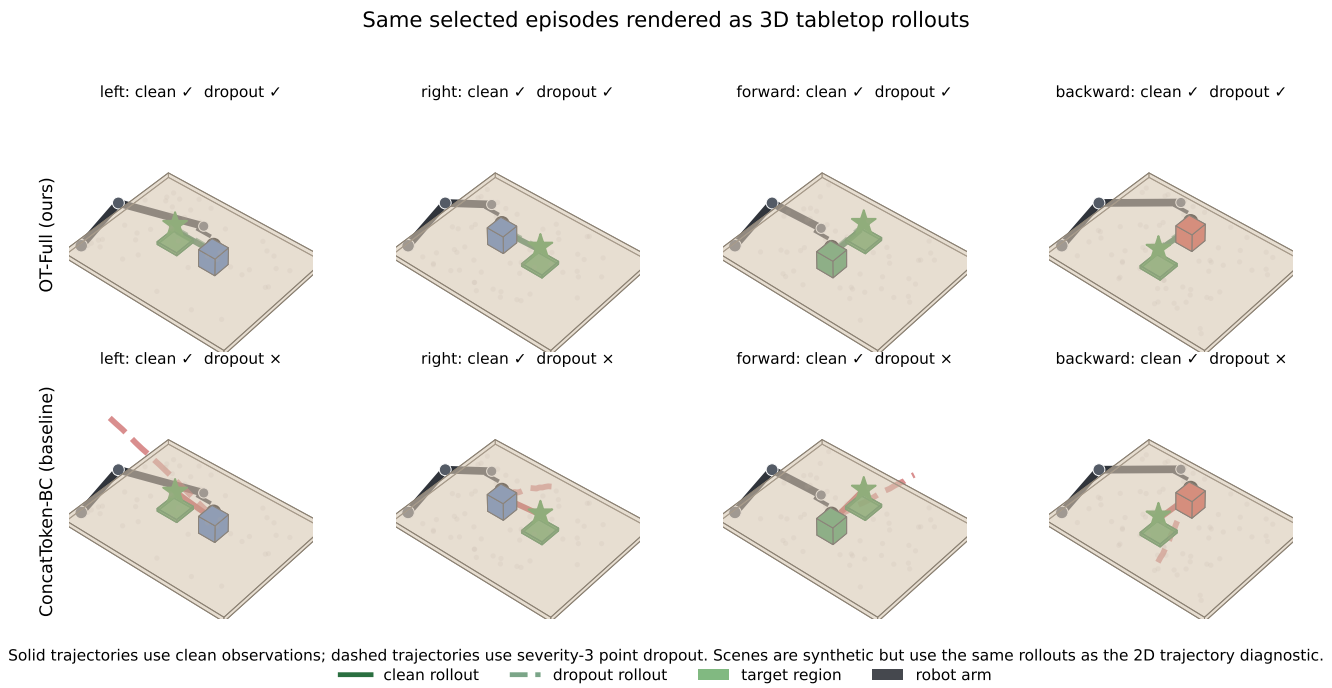

Fig. 4. Selected 3D tabletop rollouts. The tan plane denotes the tabletop coordinate surface on which the commanded object and target are placed; it is not meant to show that the robot arm is supported by the plane. The colored box is the commanded object, the green patch/star is the target goal, gray points are clutter/observed 3D points, and the dark links show a simulated robot arm for context. The arm marks the manipulation setting but is not a physically simulated gripper making contact. Columns are left, right, forward, and backward language commands. The plotted scene visualizes the object trajectory from start to target; the policy input is two camera-offset point-cloud views, not the rendered figure itself. The policy outputs short translation chunks for the object, and the rollout applies those translations kinematically rather than simulating contact-rich pushing or grasping. Dropout removes observed 3D points but leaves the task and state fixed. Solid curves are clean rollouts; dashed curves are dropout rollouts. The proposed OT barycenter tokens keep the object trajectory on target, while CONCATTOKEN-BC succeeds clean but fails after dropout.

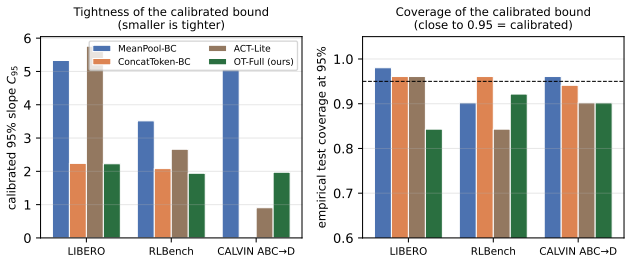

Fig. 5. Certificate audit. Left: calibrated 95 percent slope, where smaller is tighter. Right: test-half empirical coverage at the 95 percent target. The proposed model gives the tightest non-degenerate bound on the real datasets while staying near the target coverage.

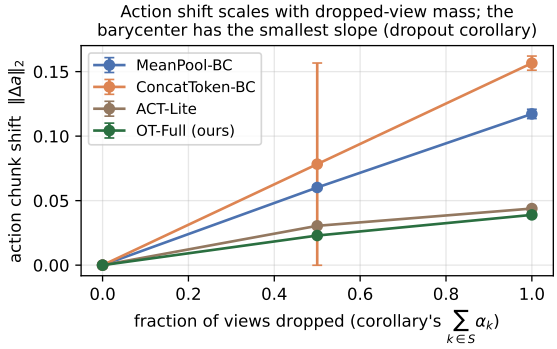

Fig. 6. Action shift as views are replaced by an all-zero missing-view dummy tensor in the diagnostic. The barycenter slope is the smallest of the four methods, consistent with the OT radius accumulating one weighted dummy-view replacement term per dropped view.

empirical measure $\mu_\emptyset$ obtained from an all-zero missing-view tensor. The observation radius contribution is therefore $\alpha_k \mathcal{S}_{\varepsilon_g}(\mu_k, \mu_\emptyset)$ for each dropped view. With zero, one, and two dropped views on synthetic, the proposed method's action shift is 0, 0.023, and 0.039. The same sweep gives 0, 0.078, and 0.157 for CONCATTOKEN-BC. The concat slope is about four times steeper. Figure 6 shows this view-dropout sweep.

The gap also persists with more optimization. On synthetic seed 7, we train the proposed model and CONCATTOKEN-BC for 600, 1,500, 5,000, and 15,000 steps. At 15,000 steps the proposed model's clean ℓ_2 is 0.0028 versus 0.0122 for CONCATTOKEN-BC, and its clean action OT is 17x smaller. The concat baseline plateaus around 5,000 steps, which points to a representation bottleneck rather than an under-trained optimizer. Figure 7 visualizes this optimization-scale sweep.

I. *Scope and audit points*

The baselines are compact matched-budget models, not large pretrained foundation models. We use this setup to isolate

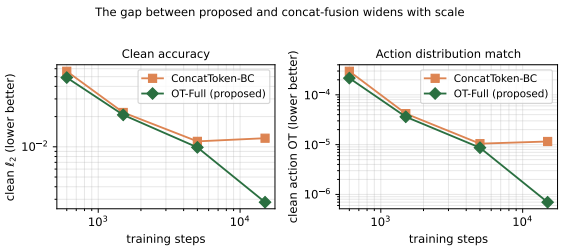

Fig. 7. Synthetic scale curve for the proposed model and CONCATTOKEN-BC. More training widens the gap instead of closing it.

the observation tokenizer and do not claim a leaderboard comparison against OpenVLA or π_0. We also record three negative audit outcomes: action z-score normalization destabilized training on synthetic and LIBERO, DiffusionPolicy-Lite did not converge at the 8,000-step synthetic horizon, and in-environment LIBERO-10 closed-loop success remained at a zero floor for all four compact models under our compute budget.

The implementation details that most affect reproducibility are the frame convention, language prior, and action distribution. RGB-D views with extrinsics are transformed before normalization; image-only observations use normalized pseudo point clouds and should be read as image-coordinate experiments. The language prior is the learned relevance softmax over points, not an object mask and not a CLIP similarity score. The barycenter support weights are fixed uniform in every reported experiment. A larger VLA stack could replace the hashed text encoder and point MLP with pretrained features, but it would still need to state these choices before a transport or shield claim is auditable.

J. Limitations and audit trail

The barycenter is not a universal robustness layer. Its ground cost is squared Euclidean distance in the learned point feature space, so a poor encoder can still make semantically different objects cheap to transport between. The bound is local and calibrated: it assumes the perturbation is inside the declared OT ball and does not protect against gross camera failure, wrong language commands, or dynamics errors after executing an action chunk. The supplementary artifact contains the CSV tables and scripts used to generate every numerical result reported in the paper. Reported closed-loop success rates use 80 episodes per condition for each of three seeds.

VII. CONCLUSION

A foundation-model robot planner needs stable 3D tokens before it can make stable plans. The Wasserstein barycentric tokenizer replaces flat multi-view token concatenation with a small differentiable transport problem. It specifies the measure being transported, uses fixed uniform barycenter weights, implements language through learned relevance weights, and gives the action head a token set with a calibrated shift bound. In matched-budget experiments, this substitution is the dominant source of the synthetic robustness gain: removing it causes a 46x corruption-stability regression, while the closed-loop concat baseline collapses under view dropout. The next step is to place this tokenizer in front of a larger pretrained VLA backbone and test whether the same bound remains useful at foundation-model scale.

References

[1] B. Zitkovich, T. Yu, S. Xu *et al.*, "RT-2: Vision-language-action models transfer web knowledge to robotic control," in *Proceedings of The 7th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 229. PMLR, 2023, pp. 2165–2183.

[2] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine, "Octo: An open-source generalist robot policy," in *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.

[3] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn, "Openvla: An open-source vision-language-action model," in *Proceedings of The 8th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, P. Agrawal, O. Kroemer, and W. Burgard, Eds., vol. 270. PMLR, 06-09 Nov 2025, pp. 2679–2713. [Online]. Available: https://proceedings.mlr.press/v270/kim25c.html

[4] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky, "π_0: A vision-language-action flow model for general robot control," 2026, arXiv: 2410.24164. [Online]. Available: https://arxiv.org/abs/2410.24164

[5] A. Brohan, N. Brown, J. Carbajal *et al.*, "RT-1: Robotics transformer for real-world control at scale," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, Jul. 2023.

[6] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, A. Tung, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Gupta, A. Wang, A. Singh, A. Garg, A. Kembhavi, A. Xie, A. Brohan, A. Raffin, A. Sharma, A. Yavary, A. Jain, A. Balakrishna, A. Wahid, B. Burgess-Limerick, B. Kim, B. Scholkopf, B. Wulfe, B. Ichter, C. Lu, C. Xu, C. Le, C. Finn, C. Wang, C. Xu, C. Chi, C. Huang, C. Chan, C. Agia, C. Pan, C. Fu, C. Devin, D. Xu, D. Morton, D. Driess, D. Chen, D. Pathak, D. Shah, D. Buchler, D. Jayaraman, D. Kalashnikov, D. Sadigh, E. Johns, E. Foster, F. Liu, F. Ceola, F. Xia, F. Zhao, F. Stulp, G. Zhou, G. S. Sukhatme, G. Salhotra, G. Yan, G. Feng, G. Schiavi, G. Berseth, G. Kahn, G. Wang, H. Su, H.-S. Fang, H. Shi, H. Bao, H. Ben Amor, H. I. Christensen, H. Furuta, H. Walke, H. Fang, H. Ha, I. Mordatch, I. Radosavovic, I. Leal, J. Liang, J. Abou-Chakra, J. Kim, J. Drake, J. Peters, J. Schneider, J. Hsu, J. Bohg, J. Bingham, J. Wu, J. Gao, J. Hu, J. Wu, J. Wu, J. Sun, J. Luo, J. Gu, J. Tan, J. Oh, J. Wu, J. Lu, J. Yang, J. Malik, J. Silverio, J. Hejna, J. Booher, J. Tompson, J. Yang, J. Salvador, J. J. Lim, J. Han, K. Wang, K. Rao, K. Pertsch, K. Hausman, K. Go, K. Gopalakrishnan, K. Goldberg, K. Byrne, K. Oslund, K. Kawaharazuka, K. Black, K. Lin, K. Zhang, K. Ehsani, K. Lekkala, K. Ellis, K. Rana, K. Srinivasan, K. Fang, K. P. Singh, K.-H. Zeng, K. Hatch, K. Hsu, L. Itti, L. Y. Chen, L. Pinto, L. Fei-Fei, L. Tan, L. J. Fan, L. Ott, L. Lee, L. Weihs, M. Chen, M. Lepert, M. Memmel, M. Tomizuka, M. Itkina, M. G. Castro, M. Spero, M. Du, M. Ahn, M. C. Yip, M. Zhang, M. Ding, M. Heo, M. K. Srirama, M. Sharma, M. J. Kim, N. Kanazawa, N. Hansen, N. Heess, N. J. Joshi, N. Suenderhauf, N. Liu, N. Di Palo, N. M. M. Shafiullah, O. Mees, O. Kroemer, O. Bastani, P. R. Sanketi, P. T. Miller, P. Yin, P. Wohlhart, P. Xu, P. D. Fagan, P. Mitrano, P. Sermanet, P. Abbeel, P. Sundaresan, Q. Chen, Q. Vuong, R. Rafailov, R. Tian, R. Doshi, R. Martin-Martin, R. Baijal, R. Scalise, R. Hendrix, R. Lin, R. Qian, R. Zhang, R. Mendonca, R. Shah, R. Hoque, R. Julian, S. Bustamante, S. Kirmani, S. Levine, S. Lin, S. Moore, S. Bahl, S. Dass, S. Sonawani, S. Song, S. Xu, S. Haldar, S. Karamcheti, S. Adebola, S. Guist, S. Nasiriany, S. Schaal, S. Welker, S. Tian, S. Ramamoorthy, S. Dasari, S. Belkhale, S. Park, S. Nair, S. Mirchandani, T. Osa, T. Gupta, T. Harada, T. Matsushima, T. Xiao, T. Kollar, T. Yu, T. Ding, T. Davchev, T. Z. Zhao, T. Armstrong, T. Darrell, T. Chung, V. Jain, V. Vanhoucke, W. Zhan, W. Zhou, W. Burgard, X. Chen, X. Wang, X. Zhu, X. Geng, X. Liu, X. Liangwei, X. Li, Y. Lu, Y. J. Ma, Y. Kim, Y. Chebotar, Y. Zhou, Y. Zhu, Y. Wu, Y. Xu, Y. Wang, Y. Bisk, Y. Cho, Y. Lee, Y. Cui, Y. Cao, Y.-H. Wu, Y. Tang, Y. Zhu, Y. Zhang, Y. Jiang, Y. Li, Y. Li, Y. Iwasawa, Y. Matsuo, Z. Ma, Z. Xu, Z. J. Cui, Z. Zhang, and Z. Lin, "Open x-embodiment: Robotic learning datasets and rt-x models : Open x-embodiment collaboration0," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 6892–6903.

[7] M. Shukor, D. Aubakirova, F. Capuano, P. Kooijmans, S. Palma, A. Zouitine, M. Aractingi, C. Pascal, M. Russi, A. Marafioti, S. Alibert, M. Cord, T. Wolf, and R. Cadene, "Smolvla: A vision-language-action model for affordable and efficient robotics," 2025, arXiv: 2506.01844. [Online]. Available: https://arxiv.org/abs/2506.01844

[8] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," in *Proceedings of Robotics: Science and Systems*, 2023.

[9] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *Int. J. Rob. Res.*, vol. 44, no. 10-11, pp. 1684–1704, Sep. 2025. [Online]. Available: https://doi.org/10.1177/02783649241273668

[10] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 77–85.

[11] Y. Pang, W. Wang, F. E. H. Tay, W. Liu, Y. Tian, and L. Yuan, "Masked autoencoders for point cloud self-supervised learning," in *Computer Vision - ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part II*. Berlin, Heidelberg: Springer-Verlag, 2022, pp. 604–621. [Online]. Available: https://doi.org/10.1007/978-3-031-20086-1_35

[12] Y. Hong, H. Zhen, P. Chen, S. Zheng, Y. Du, Z. Chen, and C. Gan, "3d-llm: injecting the 3d world into large language models," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, ser. NIPS '23. Red Hook, NY, USA: Curran Associates Inc., 2023.

[13] R. Xu, X. Wang, T. Wang, Y. Chen, J. Pang, and D. Lin, "Pointllm: Empowering large language models to understand point clouds," in *Computer Vision - ECCV 2024*, A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, and G. Varol, Eds. Cham: Springer Nature Switzerland, 2025, pp. 131–147.

[14] H. Zhen, X. Qiu, P. Chen, J. Yang, X. Yan, Y. Du, Y. Hong, and C. Gan, "3d-vla: a 3d vision-language-action generative world model," in *Proceedings of the 41st International Conference on Machine Learning*, ser. ICML'24. JMLR.org, 2024.

[15] X. Li, L. Heng, J. Liu, Y. Shen, C. Gu, Z. Liu, H. Chen, N. Han, R. Zhang, H. Tang, S. Zhang, and H. Dong, "3ds-vla: A 3d spatial-aware vision language action model for robust multi-task manipulation," in *Proceedings of The 9th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, J. Lim, S. Song, and H.-W. Park, Eds., vol. 305. PMLR, 27-30 Sep 2025, pp. 2344–2359. [Online]. Available: https://proceedings.mlr.press/v305/li25g.html

[16] L. Kong, Y. Liu, X. Li, R. Chen, W. Zhang, J. Ren, L. Pan, K. Chen, and Z. Liu, "Robo3d: Towards robust and reliable 3d perception against corruptions," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 19 937–19 949.

[17] H. Liu, J. Jia, and N. Z. Gong, "Pointguard: Provably robust 3d point cloud classification," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 6182–6191.

[18] H. Liu, S. Ruan, J. Long, J. Wu, J. Hou, H. Tang, T. Jiang, W. Zhou, and W. Yao, "Eva-vla: Evaluating vision-language-action models' robustness under real-world physical variations," 2026, arXiv: 2509.18953. [Online]. Available: https://arxiv.org/abs/2509.18953

[19] Q. Gu, Y. Ju, S. Sun, I. Gilitschenski, H. Nishimura, M. Itkina, and F. Shkurti, "SAFE: Multitask failure detection for vision-language-action models," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. [Online]. Available: https://openreview.net/forum?id=XPyAukgsFf

[20] M. Cuturi, "Sinkhorn distances: lightspeed computation of optimal transport," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13. Red Hook, NY, USA: Curran Associates Inc., 2013, pp. 2292–2300.

[21] G. Peyré and M. Cuturi, "Computational optimal transport," *Found. Trends Mach. Learn.*, vol. 11, no. 5-6, pp. 355–607, Feb. 2019. [Online]. Available: https://doi.org/10.1561/2200000073

[22] J. Feydy, T. Séjourné, F.-X. Vialard, S.-i. Amari, A. Trouvé, and G. Peyré, "Interpolating between optimal transport and MMD using sinkhorn divergences," in *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, vol. 89. PMLR, 2019, pp. 2681–2690.

[23] M. Agueh and G. Carlier, "Barycenters in the wasserstein space," *SIAM Journal on Mathematical Analysis*, vol. 43, no. 2, pp. 904–924, 2011.

[24] M. Cuturi and A. Doucet, "Fast computation of wasserstein barycenters," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 2. Bejing, China: PMLR, 22-24 Jun 2014, pp. 685–693. [Online]. Available: https://proceedings.mlr.press/v32/cuturi14.html

[25] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, "Libero: benchmarking knowledge transfer for lifelong robot learning," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, ser. NIPS '23. Red Hook, NY, USA: Curran Associates Inc., 2023.

[26] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[27] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard, "Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7327–7334, 2022.

Appendix A
Algorithms

Algorithm 1 Differentiable barycentric tokenizer

Require: Views $\{u_{k,i}\}$, instruction ℓ, learned query supports $\{b_m^0\}_{m=1}^M$

Encode text h_ℓ with the hashed-token text encoder.
for each view k and point i **do**
  Encode feature $x_{k,i} \leftarrow \phi_\psi(u_{k,i}) + W_\ell h_\ell$.
  Compute relevance logit $s_{k,i} \leftarrow g_\psi([\phi_\psi(u_{k,i}), W_\ell h_\ell])$.
end for
Set view weights $w_{k,i} \leftarrow \mathrm{softmax}_i(s_{k,i})$ and barycenter weights $r_m \leftarrow 1/M$.
Initialize $b_m \leftarrow \mathrm{LayerNorm}(b_m^0)$.
for $q = 1, \ldots, T_B$ **do**
  for $k = 1, \ldots, K$ **do**
    $C_{mi}^{(k)} \leftarrow \|b_m - x_{k,i}\|_2^2$.
    $P^{(k)} \leftarrow \mathrm{Sinkhorn}(C^{(k)}, r, w_k, \varepsilon_g)$.
    for $m = 1, \ldots, M$ **do**
      $z_m^{(k)} \leftarrow M \sum_i P_{mi}^{(k)} x_{k,i}$.
    end for
  end for
  for $m = 1, \ldots, M$ **do**
    $\bar{z}_m \leftarrow \sum_k \alpha_k z_m^{(k)}$.
    $b_m \leftarrow \mathrm{LayerNorm}(\mathrm{GRU}(\bar{z}_m, b_m))$.
  end for
end for
return barycentric tokens $\{b_m\}_{m=1}^M$.

Algorithm 2 Runtime Wasserstein margin shield

Require: Candidate chunks $\{a^{(r)}\}_{r=1}^R$, declared radius ρ, budget κ, calibrated slope C_γ, intercept b_γ, clean-cost estimator $\widehat{J}$

$\Delta_\gamma \leftarrow C_\gamma \rho + b_\gamma$.
for candidate $r = 1, \ldots, R$ **do**
  Compute predicted clean cost $\widehat{J}_r$.
  Compute margin $\Gamma_r \leftarrow \kappa - \widehat{J}_r - \Delta_\gamma$.
end for
if $\max_r \Gamma_r \geq 0$ **then**
  Execute the first action of the feasible candidate with largest margin.
else
  Execute the low-velocity fallback action.
end if

Appendix B
Dataset and preprocessing details

Appendix C
Additional synthetic diagnostics

Appendix D
Reproducibility details

The anonymous code release contains training, evaluation, ablation, predecode, and figure scripts under `codes/scripts/`. The OT primitives are in `codes/src/ot_robust3dvla/ot.py`; the point encoder, barycentric tokenizer, and policy heads are in `codes/src/ot_robust3dvla/models.py`. Real datasets are predecoded once to per-episode HDF5 files to avoid video-loader instability. Reported seeds are $\{7, 11, 19\}$, and all experiments were run on a single NVIDIA RTX 5090.

TABLE VII

DATASET AND PREPROCESSING DETAILS USED BY THE MATCHED-BUDGET RUNS. WINDOWS ARE CAPPED BY THE LOCAL REGISTRY AND SPLIT 85/15 BY A SEEDED RANDOM SPLIT; EVALUATION USES THE HELD-OUT SPLIT, CAPPED AT 20 BATCHES.

Dataset	suite/tasks	train	eval	views	point representation
Synthetic	tabletop object translation; left/right/forward/backward commands	6963	1229	2	camera-offset XYZ/RGB, 512 points/view
LIBERO	predecoded LIBERO demonstrations; LIBERO-10 only for the closed-loop zero-floor audit	17000	3000	2	normalized image-coordinate pseudo XYZ/RGB, 1024 points/view
RLBench	predecoded RLBench LeRobot-format training windows	2550	450	2	normalized image-coordinate pseudo XYZ/RGB, 1024 points/view
CALVIN ABC→D	ABC train, D evaluation windows from the LeRobot split	2550	450	2	normalized image-coordinate pseudo XYZ/RGB, 1024 points/view

TABLE VIII

SYNTHETIC DIAGNOSTICS SUPPORTING THE GEOMETRIC-TOKENIZER AND TOKEN-COUNT CLAIMS. LOWER CLEAN ℓ_2 AND LOWER $\bar{S}_\varepsilon$ ARE BETTER. FPS: FARTHEST-POINT SAMPLING.

Diagnostic	setting	clean ℓ_2	$\bar{S}_\varepsilon \times 10^6$	latency ms
Tokenizer	Concat	–	101.9 ± 24.5	–
Tokenizer	FPS	–	8.65 ± 0.99	–
Tokenizer	barycentric transport	–	2.38 ± 0.16	–
Token count	$M = 4$	0.0458	2.12	0.35
Token count	$M = 8$	0.0429	1.91	0.24
Token count	$M = 16$	0.0495	4.38	0.93
Token count	$M = 32$	0.0435	2.13	0.58
Token count	$M = 64$	0.0406	2.84	0.47