SEVERITY TARGETS CAN REWARD CORRUPTED LABELS: DIAGNOSING SAMPLE WEIGHTS FOR CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

A difficult trajectory is not necessarily a reliable one. We show that severity-based sample weighting can conflate these properties and reward corrupted action labels. With 20% of Push-T training episodes corrupted, applying a severity target directly gives corrupted windows $1.89\times$ the clean-window weight; a learned weighting network retains the same preference. A direct-target control and a feature $\times$ target ablation locate the failure in the target rather than the network. Replacing severity-only supervision with a consistency-aware target changes the corrupted-versus-clean ordering, while an amplitude-matched control that preserves the original ordering does not recover performance. A jerk- and path-matched negative control confirms the consistency-aware target preserves genuinely hard but correct maneuvers while collapsing a matched corrupted twin, so it reads label reliability rather than dynamical roughness. Across four tested synthetic corruption modes, the effect follows the target components: corruptions that inflate action jerk or path length are up-weighted, whereas a constant offset that leaves them intact is nearly neutral. Five-seed Push-T experiments further show that target correlation and closed-loop utility need not agree. USV experiments provide complementary calibration diagnostics and the same separation between target agreement and task utility. We therefore recommend evaluating weight ordering, dispersion, corruption preference, and matched task outcomes separately.

1 INTRODUCTION

Offline control data mix at least two properties that a weighting rule should not confuse. *Severity* describes disturbance or dynamical stress; *reliability* describes whether an action label is consistent with the observed trajectory. Hard-example mining and learned reweighting often prioritize difficult samples (Bengio et al., 2009; Shrivastava et al., 2016; Lin et al., 2017; Ren et al., 2018; Shu et al., 2019), but a severe valid trajectory and a corrupted label call for opposite treatment. Noisy-label methods make the same conceptual distinction in classification (Jiang et al., 2018; Han et al., 2018).

We study what happens when a sample-weight network is calibrated to an explicit severity target. The central question is not only whether the network fits that target, but whether the target orders the data correctly. Our evidence follows four steps. First, severity is not reliability: synthetic action corruptions can increase severity statistics while reducing label consistency. Second, the target causes the resulting preference: applying it directly reproduces the corrupted-sample up-weighting, and adding a consistency feature while retaining the severity target does not remove it. Third, a consistency-aware target changes the corrupted-versus-clean ordering. Fourth, an amplitude-matched control preserves the severity ordering while matching the corrected target's dispersion and fails to recover performance, isolating ordering rather than weight magnitude. Fifth, a jerk- and path-matched negative control shows the consistency-aware target keeps genuinely hard but correct maneuvers up-weighted while collapsing a matched corrupted twin, so the correction reads reliability rather than dynamical roughness.

Push-T is the primary test bed because it provides public trajectories, clean closed-loop evaluation, and controlled action corruption. We use a full feature $\times$ target ablation, direct-target controls, four corruption modes, and leave-one-component-out attribution. The tested modes show that the failure is component-specific: jerk- and path-inflating corruptions are rewarded, whereas a constant offset that leaves those statistics nearly unchanged is not. These boundary results use three seeds

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

outside the five-seed Gaussian study and are treated as exploratory mechanism evidence rather than a population-level robustness claim.

USV experiments play a complementary diagnostic role. They show that explicit alignment produces seed-stable target agreement and measurable within-severity weight structure, while task metrics can select a different operating point. Comparisons with Meta-Weight-Net and L2RW distinguish explicit target fitting from validation-meta-gradient weighting. Together, the results motivate reporting weight ordering, dispersion, seed stability, corruption preference, and matched task outcomes as separate quantities (Appendix A).

2 Related Work

World models and embodied predictive-control systems use latent dynamics, diffusion, and flow-based sequence predictors for planning and visuomotor control (Ha & Schmidhuber, 2018; Hafner et al., 2018; 2019; 2023; Ho et al., 2020; Lipman et al., 2022; Chi et al., 2023; Liu et al., 2024; Team et al., 2024; Black et al., 2024). Their training data vary in disturbance severity, coverage, and difficulty, yet the loss is averaged approximately uniformly across windows. We keep the model and policy architectures fixed and study how supervision can be redistributed within such an offline dataset.

Learning-to-reweight and Meta-Weight-Net infer example weights through validation signals (Ren et al., 2018; Shu et al., 2019). Curriculum learning, focal loss, online hard-example mining, and importance sampling change optimization pressure through other measures of difficulty or utility (Bengio et al., 2009; Lin et al., 2017; Shrivastava et al., 2016; Katharopoulos & Fleuret, 2018). Influence functions (Koh & Liang, 2017), gradient tracing (Pruthi et al., 2020), and data attribution methods (Ilyas et al., 2022; Park et al., 2023) analyze how training examples affect predictions. Our experiments examine the weights themselves: their dispersion, agreement with a specified severity target, and stability across seeds. The meta-learned baselines optimize a validation objective rather than this target, allowing us to compare task-driven weighting with explicit severity calibration. Noisy-label methods such as MentorNet and co-teaching already separate difficulty from reliability (Jiang et al., 2018; Han et al., 2018); our contribution is a controlled dissociation in embodied control—showing the conflation is component-specific (tracking inflated dynamical statistics, not label validity), traceable to the target rather than the network via direct-target and feature $\times$ target ablations, and visible in closed-loop success.

Robust imitation learning often broadens the data distribution, injects perturbations, or collects corrective supervision online (Ross et al., 2011; Laskey et al., 2017). Those methods change data collection, whereas our setting assumes a fixed offline corpus. Residual severity-aware weighting changes only the training loss and can be added to an existing world-model and policy stack.

3 Problem Setup

3.1 Data and Perturbation Signals

Each sample is a window $s_i = (h_i^{\mathrm{wave}}, h_i^{\mathrm{state}}, f_i^{\mathrm{wave}}, u_i, g_i, m_i)$ with past wave history, state history, future wave target, control sequence, goal, and sea-state metadata. The metadata m_i provides a severity proxy for training and analysis.

3.2 Importance Target

The importance network is supervised by a batch-relative target built from perturbation intensity and control stress. We use calibration to mean agreement with this specified target, measured by weight–target correlation and within-severity dispersion. Task utility is evaluated separately using prediction errors and control outcomes.

4 METHOD

4.1 ARCHITECTURE

The method has three learned modules: a diffusion wave predictor g_ψ, a dual-memory flow policy π_ω, and an importance network f_θ. The importance network is a 3-layer MLP with GELU activations and hidden width 128. Table 8 (Appendix D.1) lists the full 17-D input. When no wave prediction is available, the last two inputs are zero. Fixed hyperparameters are listed in Appendix B (Table 6).

4.2 CALIBRATION TARGET

For each sample we build a calibration target y_i from an 8-D wave-stage vector: sea-state scale, goal magnitude, wave-history energy and variability, future-wave energy, prediction error, speed variance, and jerk. The policy stage appends a detached mean per-step policy loss. Components are standardized within each minibatch, averaged, shifted to be positive, and normalized to unit mean (Appendix B.2). Future-wave labels and prediction errors enter this training-time target; the weighting network receives the 17-D feature vector in Table 8.

The residual module outputs a raw score r_i, which is converted into a positive multiplier with a fixed temperature and clipping range:

$$\tilde{w}_i = \mathrm{clip}(\mathrm{softplus}(r_i/T) + 10^{-3}, w_{\min}, w_{\max}), \qquad T = 1.0, \quad w_{\min} = 0.25, \quad w_{\max} = 4.0.$$

The stage-specific base prior is a sea-state lookup combined with a relative-energy difficulty factor q_i (Appendix B.2):

$$b_i = \beta_{s(i)} q_i, \qquad \beta_{\mathrm{calm}} = 1.0, \beta_{\mathrm{moderate}} = 1.1, \beta_{\mathrm{rough}} = 1.3, \beta_{\mathrm{storm}} = 1.6.$$

The final residual weight is

$$\hat{w}_i = \frac{b_i \tilde{w}_i}{\frac{1}{B}\sum_j b_j \tilde{w}_j}.$$

The sea-state prior sets a coarse emphasis, and the residual multiplier adjusts individual windows. Setting all $\beta_s = 1$ gives the `uniform_resid` control: the sea-state lookup becomes uniform, while the relative-energy factor q_i remains. Thus `uniform_resid` is distinct from plain uniform weighting. Algorithm 1 gives the training procedure.

4.3 WEIGHTED TRAINING OBJECTIVE

For the wave stage, the per-sample loss is the diffusion noise-prediction error $\ell_i^{\mathrm{wave}} = \|\epsilon_\psi - \epsilon\|_2^2$. For the policy stage, the main loss is the flow-matching error over action trajectories. The total objective uses detached weights in the task term and a fixed alignment weight $\lambda_{\mathrm{align}} = 0.5$:

$$\mathcal{L}_{\mathrm{stage}} = \frac{1}{B}\sum_{i=1}^{B} \mathrm{stopgrad}(\hat{w}_i)\ell_i + \lambda_{\mathrm{align}}\mathcal{L}_{\mathrm{align}}, \qquad \mathcal{L}_{\mathrm{align}} = \frac{1}{B}\sum_{i=1}^{B}(\hat{w}_i - y_i)^2.$$

Detaching the weights makes alignment the only gradient signal for the importance-network parameters θ. With $\lambda_{\mathrm{align}} = 0$, the network receives no learning signal from the stated objective. The `noalign` condition therefore measures the effect of an untrained residual function, including its initialization-dependent ordering. The meta-gradient baselines in Section 6.4 retain a separate training signal for their weights.

4.4 TRAINING PROCEDURE

Training proceeds in two stages. We first train the wave predictor and importance network using the weighted task loss and calibration target. We then freeze the wave predictor and train the policy with the residual weighting rule. Clipping and batch-mean normalization are applied in both stages, and the checkpoint with the lowest validation loss is exported.

5 Experimental Protocol

5.1 Corpus and Splits

We use a 37-episode USV corpus with stratified episode holdout, totaling 44,371 windows (38,326 train / 6,045 validation). Splits are indexed by episode offset: the primary split provides the training pool and validation set used throughout, and a held-out shifted split (offset-2) is reserved for the forward calibration diagnostics.

5.2 Baseline Suite

The USV experiments use two training groups on the same 37-episode benchmark. The original group compares `uniform`, the heuristic `imbalance` rule, `hard_mining`, and learned weighting. The residual group compares `uniform_resid`, `balanced_resid`, `mild_storm`, `storm_focus`, `prior_only`, and `noalign`. The prior-only control fixes the residual multiplier to 1; no-align removes the residual network's alignment objective. Task comparisons are made within each training group. Tables 9 and 7 specify the controls and their display names.

Two alignment-ablation pairs use separate training protocols. The pair (`uniform_resid`, `noalign_matched`) supplies the mixed-split results and five-seed reruns in Table 2. The pair (`aligned`, `noalign`) supplies the forward diagnostics in Tables 11 and 12. We use the first pair to measure changes in matched task errors and seed stability, and the second to examine within-severity structure and split shift.

5.3 Metrics

We report wave prediction MSE and MAE, policy full-sequence MSE, rolling MSE, and control smoothness. Rolling MSE measures the action error at the steps selected by receding-horizon execution: the policy is queried at each step and its first action is compared with the target. Aggregate rolling MSE averages across sea states; storm rolling MSE restricts evaluation to the storm bucket. Full-sequence MSE evaluates the entire predicted action chunk. The fixed-split seed study uses the same 6,045 validation windows and reports means and sample standard deviations over five seeds. Push-T supplies the closed-loop success and coverage evaluations.

6 Auxiliary Calibration Diagnostics on USV

The USV study establishes that explicit calibration produces structured, seed-stable weights and illustrates why target agreement must be separated from task utility. It is supporting evidence for the Push-T corruption mechanism in Section 7.1, not an independent claim that severity weighting improves control.

6.1 Baseline Comparison

In the original training group, uniform weighting gives the lowest aggregate wave, full-sequence, and rolling errors (Table 1). In the residual group, `uniform_resid` has the lowest wave and full-sequence errors, while `storm_focus` has the lowest aggregate rolling error. The two policy metrics therefore select different residual configurations.

6.2 Controlled Decomposition

Increasing storm emphasis from `uniform_resid` to `storm_focus` reduces aggregate rolling MSE from 0.399369 to 0.379460 while full-sequence MSE rises from 0.078265 to 0.097478, with intermediate configurations following the same ordering (Figure 6). Fixing the residual multiplier to 1 gives `prior_only`, whose mixed-split wave, full-sequence, and rolling errors exceed those of `uniform_resid`, which also has lower policy MSE on the fixed and offset-2 forward diagnostics (Table 10); this isolates the combined change from the prior-only to the calibrated residual configuration.

Table 1: USV mixed-split results. Each panel is a separate training group. These single-run values are descriptive; the matched five-seed comparison is reported in Table 2. The original group's uniform baseline uses equal weights; the residual group's `uniform_resid` uses a learned multiplier with uniform sea-state coefficients.

Run	Wave MSE	Full-seq MSE	Rolling MSE	Smoothness
Optimization family (original mixed-split recipe)				
Uniform baseline	1.006744	0.131706	0.383866	0.050062
Hard-mining baseline	1.070183	0.134811	0.384610	0.051008
Imbalance baseline	1.101341	0.138619	0.393550	0.050655
Learned weighting	1.164405	0.141111	0.391984	0.050778
Residual family (learned-weighting recipe, varying storm emphasis)				
uniform_resid	0.800950	0.078265	0.399369	0.035687
balanced_resid	0.839422	0.084297	0.395967	0.037340
mild_storm	0.835939	0.092522	0.381321	0.037746
storm_focus	0.832208	0.097478	0.379460	0.040525

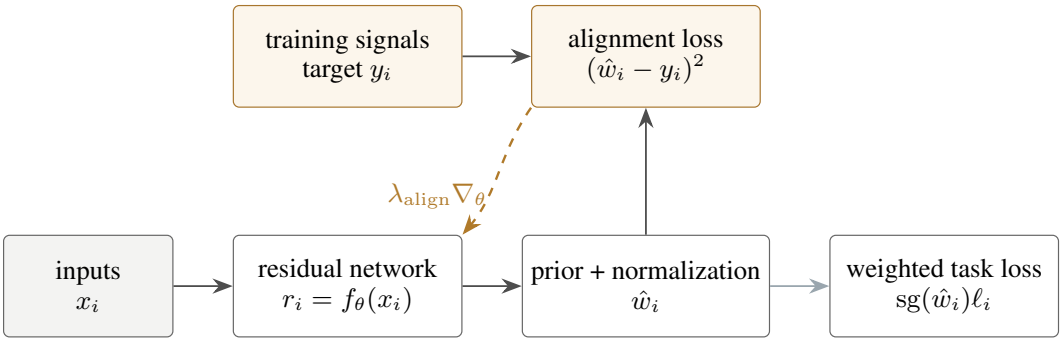

Figure 1: Gradient paths in residual-weight training. The task loss uses detached weights (grey path); the alignment loss trains the residual network (amber path). The displayed weight includes the prior, bounded multiplier, and batch normalization. Setting $\lambda_{\text{align}} = 0$ removes the network's sole learning signal.

Removing alignment increases all three mixed-split errors for the matched pair in Table 2: weight–target correlation falls from 0.600 to -0.087 on the fixed split and from 0.633 to 0.327 on offset-2, the shifted-split no-align correlation staying positive but below the calibrated model (Figure 5). The residual sweep also changes forward policy error (Table 13; Figure 7): relative to `uniform_resid`, `storm_focus` raises policy MSE by $1.196\times$ and $1.283\times$ on the fixed and offset-2 splits, accompanying the lower aggregate rolling MSE; storm-bucket results appear below.

6.3 Sea-State Behavior, Split Stability, and Calibration Structure

Within the original group, learned weighting reduces storm rolling MSE from 0.680057 to 0.647966 (4.7% over uniform) while increasing errors in the other three sea states (Table 14; Table 15); within the residual group, `storm_focus` reduces rolling error in all four buckets relative to `uniform_resid` at a higher full-sequence error, and storm error is non-monotonic in emphasis (`mild_storm` 0.686329 vs. `storm_focus` 0.695319), so aggregate and storm-bucket errors select different settings. Across five fixed-split seeds, calibrated residual weighting gives mean rolling MSE 0.417071 ± 0.005180 versus 0.426551 ± 0.008642 for no-align, with weight–target correlation in $[0.578, 0.593]$ against an untrained-control range of $[-0.261, +0.200]$ (about thirty times the standard deviation), so alignment produces consistent target agreement across initializations (Table 2; Figure 8). The independently trained forward-diagnostic pair further shows within-bucket structure: after subtracting each bucket's mean weight the aligned model retains correlations $0.612/0.633$ on the fixed/offset-2 splits versus $-0.280/-0.227$ for no-align (Table 11). In that pair, however, no-align has lower wave and policy MSE despite its negative target correlation (Table 12), unlike

Table 2: Matched alignment ablation. Top: mixed-split task errors and forward weight–target correlations for one trained pair. Bottom: fixed-split reruns, mean ± sample standard deviation over five seeds. The no-align residual network receives no learning signal from the training objective.

Method	Mixed wave	Mixed full-seq	Mixed rolling MSE	Fixed corr.	Offset-2 corr.
uniform_resid	0.800950	0.078265	0.399369	0.599624	0.632509
noalign (matched)	0.818077	0.093782	0.420892	-0.087189	0.326587

Method	Mean rolling MSE (5 seeds)	Rolling MSE range	Weight–target corr. (5 seeds)	Corr. range
uniform_resid	0.417071 ± 0.005180	[0.410406, 0.423430]	0.585527 ± 0.006267	[0.578131, 0.593317]
noalign (matched)	0.426551 ± 0.008642	[0.418574, 0.436564]	-0.100415 ± 0.186296	[-0.261284, 0.200132]

Table 3: Weight diagnostics for external USV baselines, averaged over seeds 11 and 42; residual controls are shown for reference. Feature-conditioned Meta-Weight-Net has positive target correlation but negligible weight dispersion. Adding alignment increases both. L2RW has high dispersion and low severity correlation.

Method	Weight input / supervision	Weight–target corr.	Within-bucket weight std.	Weight range
MWN	loss / meta-gradient only	−0.00	0.00002	0.0002
MWN + severity features	17-D features + loss / meta-gradient only	+0.29	0.00005	0.0009
MWN + features + calibration	17-D features + loss / meta-gradient + alignment	+0.41	0.220	3.25
L2RW	per-batch clamped meta-gradient	+0.01	1.80	35.2
Residual (aligned, ours)	17-D features / alignment	+0.59	0.143	—
Residual (no-align, ours)	17-D features / none	−0.10	0.017	—

the mixed-split matched pair—another instance of target agreement and task utility diverging (Figure 10).

6.4 External Meta-Learned Reweighting Baselines

We next compare two methods that train weights through a different objective: Meta-Weight-Net (Shu et al., 2019) and learning-to-reweight (L2RW) (Ren et al., 2018). Both use a one-step validation meta-gradient on held-out meta episodes from the same corpus, backbone, and fixed episode split. Neither receives the severity target. The `mwn_feat` control adds the residual network's 17-D severity features to the loss input; `mwn_feat_align` further adds calibration alignment to the meta objective. The latter pair holds the input and weighting architecture fixed while changing the supervision.

Meta-Weight-Net returns nearly constant weights (within-bucket standard deviation 0.00002); adding severity features raises correlation to 0.29 but leaves the standard deviation at 0.00005 and range at 0.0009, so positive correlation alone does not imply an appreciable change in training emphasis, whereas adding alignment raises the standard deviation to 0.220, the range to 3.25, and correlation to 0.41 (Table 3; Figure 15). L2RW instead has standard deviation 1.80, range 35.2, and severity correlation 0.01: its validation meta-gradient allocates nonuniform weights without fitting the chosen target. These comparisons distinguish near-constant, dispersed-but-uncalibrated, and explicitly calibrated weighting; task outcomes determine which is useful.

7 Primary Push-T Study

7.1 Clean-Data Reference

We repeat the comparisons on public LeRobot Push-T keypoint trajectories using an observation-conditioned action-chunk flow-matching transformer and closed-loop evaluation in gym-pusht. Each arm uses five training seeds and 200 evaluation episodes per model (Appendix D.6). The aligned residual network reaches correlation 0.588 ± 0.051 and within-bucket standard deviation 0.225 ± 0.006, compared with 0.054 ± 0.235 and 0.030 ± 0.002 for no-align. Meta-Weight-Net again produces nearly constant weights; its feature-and-calibration variant reaches correlation 0.781 ± 0.017.

Table 4: Push-T weight diagnostics and closed-loop outcomes. Means ± sample standard deviations use five training seeds, with 200 evaluation episodes per model. Meta-Weight-Net has the highest success mean; uniform has the highest coverage mean. Calibrated models have stronger target agreement but lower mean success than uniform.

Arm	Weight–target corr	Within-bucket std	Weight range	Success	Coverage
`uniform`	— (constant)	0.000	0.000	0.209 ± 0.032	0.834
`aligned`	$+0.588 \pm 0.051$	0.225 ± 0.006	2.08	0.194 ± 0.025	0.809
`noalign`	$+0.054 \pm 0.235$	0.030 ± 0.002	0.24	0.212 ± 0.058	0.823
`mwn`	-0.112 ± 0.603	$\leq 3 \times 10^{-5}$	$\sim 2 \times 10^{-4}$	0.215 ± 0.065	0.818
`mwn_align`	$+0.781 \pm 0.017$	0.247 ± 0.007	2.65	0.107 ± 0.033	0.757
`l2rw`	$+0.196 \pm 0.017$	2.119 ± 0.049	29.2	0.111 ± 0.038	0.741

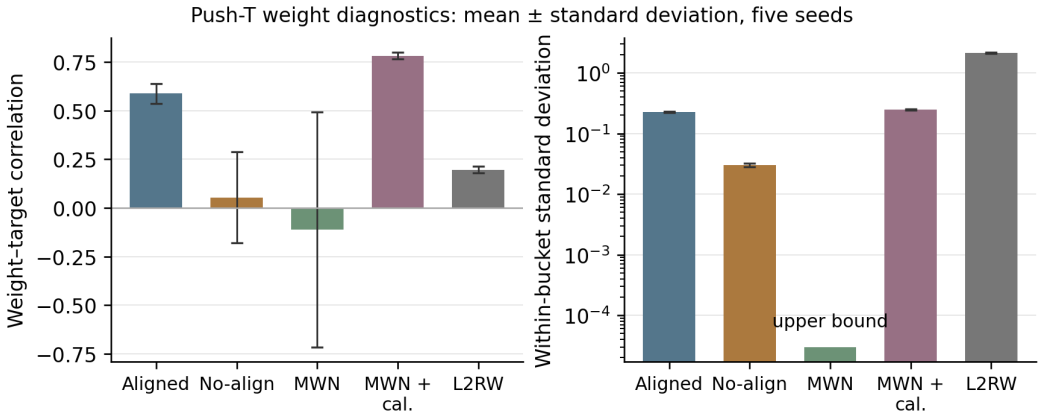

Figure 2: Push-T weight diagnostics over five seeds (mean ± sample standard deviation). Calibrated models have stronger target agreement; L2RW has high dispersion and a weaker positive correlation. Meta-Weight-Net dispersion is shown at the reported upper bound of 3×10^{-5}. Task outcomes are in Table 4.

L2RW has the largest weight range, 29.2, and within-bucket standard deviation 2.119, with a positive but weaker correlation of 0.196 ± 0.017 (Table 4). Figure 2 compares the distributions. The broad differences in dispersion also appear on the USV, while the target correlations differ between domains.

Closed-loop success does not follow the correlation ranking. Mean success is 0.209 for uniform, 0.212 for no-align, and 0.215 for Meta-Weight-Net, compared with 0.194 for aligned residual weighting and 0.107 for calibrated Meta-Weight-Net. The apparently low success values should be read together with coverage: uniform reaches mean maximum coverage 0.834 but satisfies the environment's thresholded success event in 0.209 of episodes. Thus many rollouts make substantial progress without crossing the success threshold. All arms use the same backbone, converged 8,000-step budget, and fixed evaluation episodes, so the study supports matched within-backbone comparisons rather than a claim of competitive Push-T performance. In the hardest initial-difficulty quartile, calibrated variants also have lower success and coverage than uniform (Appendix D.7). Stronger target agreement is therefore not accompanied by higher mean closed-loop performance.

7.2 Corruption Mechanism: When Severity Targets Reward Bad Labels

We inject offset-space action noise into 20% of the Push-T training episodes, keeping meta, validation, and evaluation data clean. The added `qalign` condition supplies a label-consistency feature and subtracts an inconsistency penalty from the calibration target (Appendix D.8). This fixed corruption fraction tests the mechanism at one operating point; it does not establish invariance to corruption prevalence or magnitude.

Severity-only calibration assigns higher weights to corrupted windows: the noisy-to-clean ratios are 1.122 for the residual model and 1.228 for calibrated Meta-Weight-Net. The consistency-aware target reverses this preference, giving 0.913 ± 0.004, while no-align remains close to equal weighting at 0.997 ± 0.004 (Table 5). Mean success is 0.181 for `qalign` and 0.191 for uniform; severity-only aligned and calibrated Meta-Weight-Net reach 0.133 and 0.102. Correcting the direction of the weights improves on these severity-only configurations without exceeding the uniform mean. Meta-Weight-Net remains nearly constant, and L2RW has high dispersion with correlation 0.185, compared with 0.196 on clean Push-T and 0.01 on the USV.

To localize the cause, we separate the weighting network from the calibration target. Used directly as the sample weight with no network, the severity target up-weights corrupted windows more strongly than the learned model (ratio 1.887 versus 1.122); the network attenuates but does not remove the preference. A 2×2 ablation over feature and target shows that the ordering follows the target: under severity supervision the ratio stays above one regardless of the feature, while consistency-aware supervision brings it to ≈ 1.00, whether learned or applied directly (Table 5). Net down-weighting (0.913) appears only when the consistency-aware target is combined with the consistency feature in this Gaussian grid; under a constant offset, the direct consistency-aware target already down-weights without that feature.

The amplitude control separates ordering from magnitude. `direct_shrink` preserves the severity ordering but shrinks deviations from one to match `direct_qual`'s dispersion (0.081). It still up-weights corrupted windows (ratio 1.35) and remains at the severity success level (0.133; indistinguishable from `direct`, p=0.37). Changing the ordering through `direct_qual` raises success by $+0.070$ over `direct` (t=3.84, p=0.019), whereas matching amplitude alone does not. The consistency-aware advantage over the amplitude control is $+0.051$ but marginal at five seeds (p=0.069), so we interpret this as evidence about the mechanism rather than a definitive task improvement. No Gaussian correction exceeds uniform mean success.

A remaining concern is whether the consistency-aware target simply down-weights all non-smooth actions rather than reading label reliability, which would penalize genuinely hard but correct maneuvers. We construct a matched negative control on clean expert data (Appendix D.8): the top-quartile clean windows by action jerk are genuine hard maneuvers (*clean-hard*), and reflecting their action chunk about the agent position yields a *flip* twin with jerk and path length preserved exactly but the commanded direction—and hence label consistency—inverted. Because the two groups share identical severity components, the severity target assigns them equal weight (1.21 each), confirming it cannot distinguish a corrupted maneuver from a valid one. The consistency-aware target instead keeps the clean-hard maneuvers up-weighted (1.19, above the clean-easy 1.12) while collapsing the matched flip twin to 0.45, a preservation gap of $+0.74$ (five seeds; Appendix D.8, Figure 17). The correction therefore reads reliability rather than dynamical roughness: it retains valid high-intensity control and removes only the matched corruption.

The additional corruption modes provide exploratory boundary evidence. Severity weighting is high for the three tested modes that inflate a target component (`direct` ratio 1.17–2.11) and near-neutral for the constant offset (1.03), which damages consistency while leaving path length and jerk nearly unchanged. The consistency-aware target produces ratios near one in the jerk-inflating modes and below one for the constant offset (Appendix D.8). Because the non-Gaussian modes use three seeds and the corruption fraction remains fixed at 20%, these results delimit a plausible component-level mechanism rather than establish robustness across corruption distributions.

7.3 Feature-Noise Stability

As feature-noise standard deviation rises from 0 to 0.40, weight–target correlation decreases from 0.564 to 0.417, while the storm-to-calm mean-weight ratio changes from 1.301 to 1.295 (Table 16; Figure 14). The coarse emphasis is stable under this perturbation even as sample-level target agreement weakens.

8 Discussion

Training signal and weight structure. The residual no-align control and the meta-gradient baselines test different questions. Removing alignment removes the only learning signal for the weight

network, so its initialization-dependent outputs are a control for trained calibration, not evidence that task-driven learning must fail. Meta-Weight-Net and L2RW instead receive validation meta-supervision, not our hand-specified severity target. They are strong alternative weighting objectives, but they are not dedicated noisy-label trajectory filters. A direct comparison with a small-loss or disagreement-based trajectory-filtering method remains necessary to position the consistency-aware target against robust-data-selection methods.

Target agreement and task utility. Calibration specifies which samples receive emphasis; it does not establish the value of that emphasis. On the USV residual sweep, lower aggregate rolling error comes at higher full-sequence error, and the forward-diagnostic pair with weaker target agreement has lower task error; on Push-T, calibrated models have higher target correlations but lower mean success than uniform. The noise-injection study localizes one reason: a severity target built from path length and jerk can treat label corruption as difficulty and up-weight it, with the feature $\times$ target ablation and direct-target control placing the cause in the target rather than the weighting machinery, and an amplitude-matched control confirming the consistency-aware recovery comes from correcting which samples are weighted rather than from smaller weights. The failure is component-specific—the target reacts to corruptions that inflate jerk or path length but is nearly neutral to a constant offset that leaves them intact—and correcting it does not by itself raise the success mean above uniform.

Evaluation protocol. These results motivate three measurements—weight distribution, agreement with the intended target, and task performance—which we pair with within-bucket dispersion, seed-stable correlation, a matched uniform baseline, and a known-corruption check. Appendix A gives the checklist, including how to distinguish an untrained-weight control from a trained alternative objective.

9 Scope and Future Validation

The severity targets are heuristic and batch-relative. Their components differ between USV and Push-T, so target correlation measures agreement with each domain's specified priority. The primary corruption study uses one 20% corruption fraction and one Gaussian magnitude; the three-seed mode comparison is exploratory. A fraction-and-magnitude sweep is needed to test whether the ordering transition persists outside this operating point. Push-T comparisons use one width-128 flow-matching backbone at a validation-plateaued $8{,}000$-step budget. Coverage is high relative to thresholded success, but a stronger public backbone or standard-policy reproduction is needed to establish that the conclusions persist at higher absolute success. We also do not yet compare against a dedicated noisy-label trajectory filter. The USV corpus is proprietary, and both control settings are simulated. These limitations bound the claim to the observed weighting mechanism and matched within-backbone comparisons, rather than state-of-the-art control or universal corruption robustness.

10 Conclusion

Severity and reliability induce different training priorities. On Push-T, corruptions that inflate jerk or path length receive larger severity weights; direct-target and feature $\times$ target controls locate this ordering in the target rather than the weighting network. Consistency-aware supervision changes the ordering, whereas matching its smaller dispersion while preserving the severity ordering does not recover performance. The tested boundary modes suggest that the failure follows the target components, while USV diagnostics show more generally that target agreement and task utility can diverge. These results support a concrete evaluation rule: inspect which samples a weighting target prioritizes, not only how accurately a network fits it or how widely its weights vary.

Reproducibility Statement

Appendix B specifies the importance-module hyperparameters, calibration target, and training algorithm. Appendix A gives the evaluation checklist. The Push-T experiments use five seeds (11, 22, 33, 44, 55); the external USV baselines use seeds 11 and 42. The USV matched reruns use five

seeds, and single-run comparisons are identified in the tables. The Push-T corpus and environment are public (LeRobot keypoints and gym-pusht), and the appendices give the full network, target, training, and evaluation settings needed to reproduce the reported numbers. Code and configuration files will be made available upon acceptance.

AI USAGE DISCLOSURE

AI-assisted tools were used for language editing, LaTeX restructuring, and consistency checks. The authors remain responsible for the technical claims, experimental results, citations, and final manuscript.

REFERENCES

Yoshua Bengio, Jerôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control, 2024.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion Policy: Visuomotor policy learning via action diffusion, 2023.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *arXiv preprint arXiv:1811.04551*, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Predicting predictions from training data. In *International Conference on Machine Learning (ICML)*, 2022.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning (ICML)*, 2018.

Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *International Conference on Machine Learning (ICML)*, 2018.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, 2017.

Michael Laskey, Jonathan Lee, Roy Fox, Anca Dragan, and Ken Goldberg. DART: Noise injection for robust imitation learning. In *Proceedings of the 1st Conference on Robot Learning*, 2017.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: a diffusion foundation model for bimanual manipulation, 2024.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. TRAK: Attributing model behavior at scale. In *International Conference on Machine Learning (ICML)*, 2023.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems*, 2019.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jiaming You, Pannag Bhatt, Karl Donat, Siddharth Gupta, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy, 2024.

A Calibration-Ablation Protocol for Learned Reweighting

The protocol separates the source of weight variation from its effect on the task. Record the training objective and gradient paths before interpreting a calibration ablation.

1. **Matched calibration ablation.** Remove the alignment term while retaining the same initialization, architecture, inputs, optimizer, data, and task-training budget. State whether another objective still trains the weights. If no learning signal remains, identify the condition as an untrained-weight control; compare with a trained alternative objective when testing the role of supervision type.
2. **Weight-level diagnostics.** Report target correlation, within-bucket standard deviation after centering each stratum, and weight range. Correlation can remain positive when variation is negligible, while substantial dispersion can coexist with weak target agreement. Describe both the ordering and its amplitude.
3. **Seed stability.** Report the distribution and sign of target correlation across training seeds, alongside task metrics. We use five seeds for the residual reruns and Push-T studies and two for the external USV baselines. State the seed count for each comparison.
4. **Matched uniform baseline.** Compare task outcomes under equal weights with the same data, architecture, training budget, and selection rule. Report aggregate and subgroup outcomes separately. An operating-point tradeoff requires an observed improvement on one metric and a cost on another; absence of an aggregate gain alone does not establish a tradeoff.

5. **Known-corruption stress test.** Inject corruption into the training pool while retaining clean evaluation data. Measure the noisy-to-clean weight ratio and downstream performance. This checks whether a target favors corrupted examples and whether changing that preference improves the task.

Keep independently trained ablation pairs separate, and identify the statistical unit used for uncertainty estimates. Report paired differences when evaluation seeds are shared. Weight-level and task-level results should be interpreted as separate measurements rather than substitutes.

B Method Details

B.1 Fixed Hyperparameters

B.2 Calibration Target Construction

For each sample, we build a calibration target from severity, difficulty, and stress terms. The wave-stage target is

$$c_i^{\text{wave}} = [s_i, g_i, e_i, a_i, \hat{e}_i, p_i, \sigma_i^{\text{spd}}, j_i] \in \mathbb{R}^8,$$

and the policy-stage target appends a detached policy stress term,

$$c_i^{\text{pol}} = [c_i^{\text{wave}}, \ell_i^{\text{proxy}}] \in \mathbb{R}^9, \qquad \ell_i^{\text{proxy}} = \text{stopgrad}(\bar{\ell}_i^{\text{policy}}),$$

where $\bar{\ell}_i^{\text{policy}}$ is the mean per-step policy loss in the current minibatch. The scalar components are

$$s_i = \frac{\text{sea_id}(i)}{3}, \qquad g_i = \frac{|g_i^{\text{heading}}|}{\pi} + \frac{1}{2}\frac{|g_i^{\text{speed}}|}{20} + \frac{1}{4}\frac{|g_i^{\text{dist}}|}{50},$$
$$e_i = \text{mean}\left((h_i^{\text{wave}})^2\right), \quad a_i = \text{std}(h_i^{\text{wave}}),$$
$$\hat{e}_i = \text{mean}\left((f_i^{\text{wave}})^2\right), \quad p_i = \text{mean}\left((\hat{f}_i^{\text{wave}} - f_i^{\text{wave}})^2\right).$$
$$\sigma_i^{\text{spd}} = \text{Var}\left(\sqrt{v_{x,t}^2 + v_{y,t}^2}\right), \qquad j_i = \text{mean}\left((\Delta v_t)^2\right).$$

For each component c_{ik}, we standardize within the batch, average the standardized terms, shift by a small floor, and renormalize:

$$\tilde{c}_{ik} = \frac{c_{ik} - \mu_k}{\sigma_k + \varepsilon}, \qquad \bar{c}_i = \frac{1}{K}\sum_{k=1}^{K} \tilde{c}_{ik}, \qquad y_i = \frac{\bar{c}_i - \min_j \bar{c}_j + 0.25}{\frac{1}{B}\sum_j (\bar{c}_j - \min_\ell \bar{c}_\ell + 0.25)}.$$

The resulting label is positive and has unit batch mean. Its value depends on the other examples in the minibatch, so it specifies a relative priority within that batch.

The base prior combines the sea-state lookup with the same difficulty factor used in the training code: $q_i = 1 + \alpha_{\text{wave}}\delta_i^{\text{wave}}$ in the wave stage and $q_i = 1 + \alpha_{\text{policy}}\delta_i^{\text{policy}}$ in the policy stage, with $\delta_i^{\text{wave}} = \max(0, \rho_i^{\text{wave}} - 1)$ and $\delta_i^{\text{policy}} = \max(0, \rho_i^{\text{policy}} - 1)$. The ratios ρ_i^{wave} and ρ_i^{policy} are each sample energy divided by the batch mean energy, using future waves in the wave stage and predicted waves in the policy stage.

B.3 Training Algorithm

C Run Glossary

Several runs appear under more than one display name across the result tables. Table 7 resolves these to a single identifier and states the training group each run belongs to; comparisons are valid only within a group.

Algorithm 1 Residual importance learning with explicit calibration target and normalization

Require: minibatch $\mathcal{B}$, base prior b_i, temperature T, bounds $[w_{\min}, w_{\max}]$, alignment weight λ_{align}

for each sample $s_i \in \mathcal{B}$ **do**
  Build feature vector $x_i \leftarrow \phi(s_i)$
  Predict raw residual score $r_i \leftarrow f_\theta(x_i)$
  Convert it to a bounded multiplier $\tilde{w}_i \leftarrow \text{clip}(\text{softplus}(r_i/T) + 10^{-3}, w_{\min}, w_{\max})$
  Form the unnormalized sample weight $u_i \leftarrow b_i \tilde{w}_i$
end for
Normalize across the batch $\hat{w}_i \leftarrow u_i / \frac{1}{B}\sum_j u_j$
Build the calibration target y_i from severity, difficulty, and stress components
Minimize $\frac{1}{B}\sum_i \text{stopgrad}(\hat{w}_i)\ell_i + \lambda_{\text{align}}\frac{1}{B}\sum_i(\hat{w}_i - y_i)^2$
Freeze the wave predictor and reuse the same weighting rule for policy training

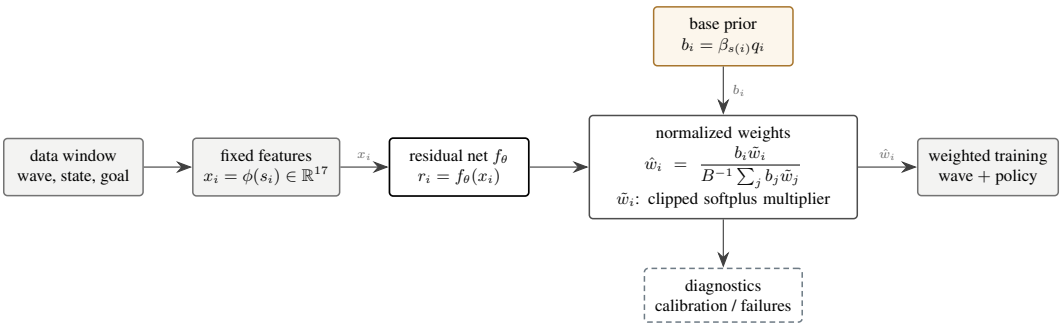

Figure 3: Residual weighting pipeline. A 17-D feature vector enters the residual network; its bounded multiplier modifies the base prior before batch-mean normalization. The resulting weights enter the wave and policy losses and are evaluated by the weight-level diagnostics. The alignment gradient is shown in Figure 1.

D Additional Results

D.1 Feature Vector and Baseline Taxonomy

D.2 Cross-Protocol Frontier and Storm Robustness

D.3 Sea-State Breakdown

D.4 Supplementary Figures

D.5 Feature-Noise Stability

D.6 Push-T Second-Domain Replication

Protocol. We use the public LeRobot Push-T keypoint corpus and the public gym-pusht environment. Observations are 18-D (2-D agent position plus eight object keypoints); the policy is an observation-conditioned action-chunk flow-matching transformer (3 layers, model width 128, 4 heads) that maps a 2-step observation history to an 8-step action chunk, executed receding-horizon 4 steps at a time with 10 Euler integration steps at inference. Episodes are split 80/10/10 into train/meta/validation pools by episode (the meta pool is used only by the Meta-Weight-Net and L2RW arms); all arms train 8,000 steps with AdamW (learning rate 3×10^{-4}, batch 256). The importance network is a 2-hidden-layer MLP (width 128) on an 8-D observable feature vector (agent and keypoint speeds, keypoint dispersion, agent–block distance, normalized positions); weights pass through a softplus, are clamped to $[0.25, 2]$, and batch-mean normalized. The USV multiplier uses the wider interval $[0.25, 4]$. The calibration target applies the batch-standardization and normalization rule of Appendix B.2 to four label-derived task-intensity components (future block displacement, action path length, action jerk, agent–block contact distance) and the detached per-sample loss, with alignment weight $\lambda = 0.5$. Meta-Weight-Net uses the standard loss-to-weight MLP

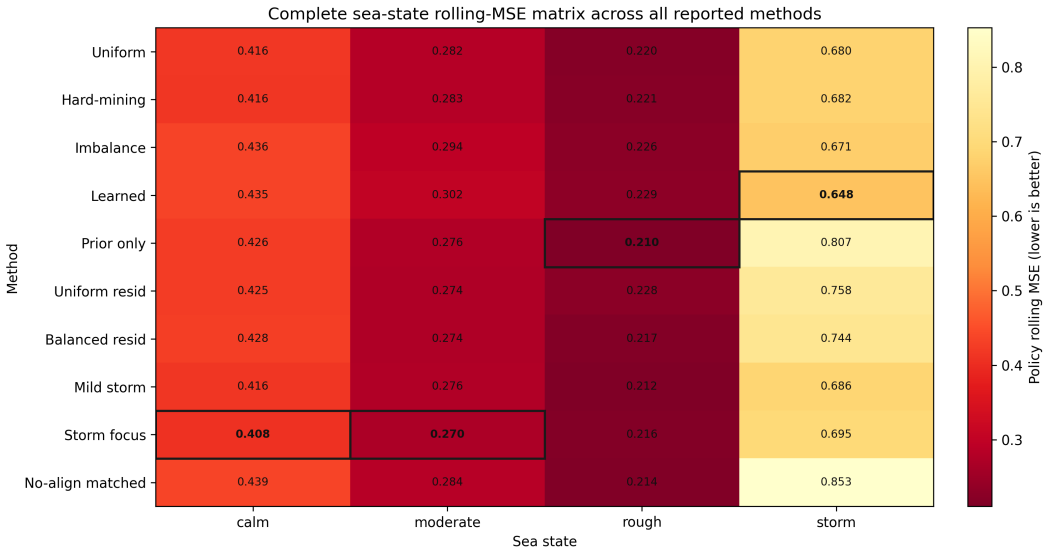

Figure 4: Sea-state rolling MSE. Rows retain their original and residual training-group assignments. The original learned model improves on its uniform baseline in storm; within the residual sweep, the lowest storm error occurs at the mild-storm setting.

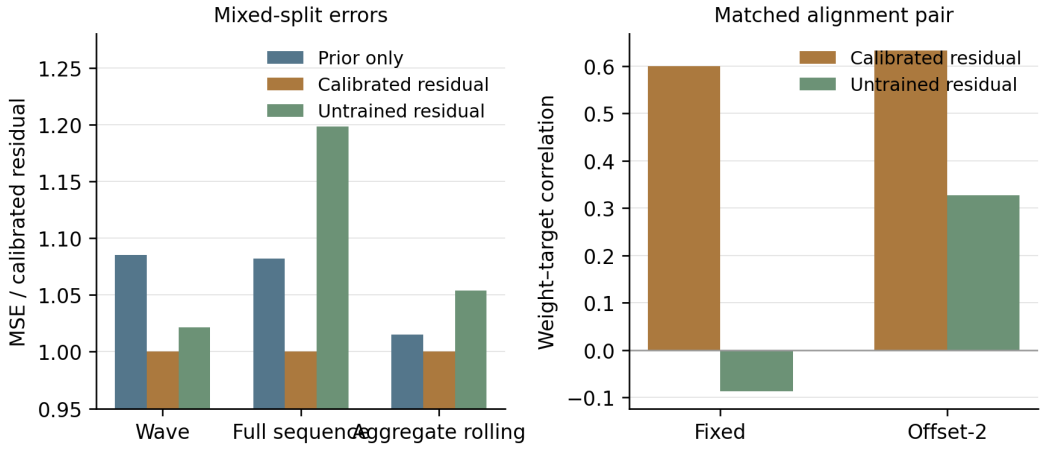

Figure 5: Matched USV controls. Left: mixed-split errors normalized by calibrated residual weighting. Right: fixed and offset-2 weight–target correlations for the matched alignment pair. The shifted no-align correlation remains positive.

(hidden width 100, meta learning rate 10^{-3}) with a one-step meta-gradient on the held-out meta episodes; `mwn_align` adds the observable features to its input and the calibration alignment term to its meta objective. The `l2rw` arm (Ren et al. 2018) sets each example weight from a one-step meta-gradient of the held-out meta loss with respect to a per-example weight ϵ_i (initialized at zero): it takes $\tilde{w}_i = \max(0, -\partial \mathcal{L}_{\text{meta}}/\partial \epsilon_i)$ and normalizes $\tilde{w}$ to unit batch mean (falling back to uniform when the batch sum is zero), with no weight network and no calibration signal, matching the USV external baseline. Difficulty buckets are quartiles of the calibration target, playing the role sea states play on the USV corpus. Closed-loop evaluation runs 200 fixed-seed episodes per model (at most 300 steps each); we report the thresholded environment success flag and maximum coverage. Each arm is trained with 5 seeds. The 8,000-step budget is chosen so that the held-out validation loss plateaus: the uniform arm's best validation MSE is 0.065 ± 0.005 across the five seeds, and the same backbone and budget are used for the clean-data replication, where uniform reaches success 0.209 and coverage 0.834. The gap between these metrics indicates substantial partial task comple-

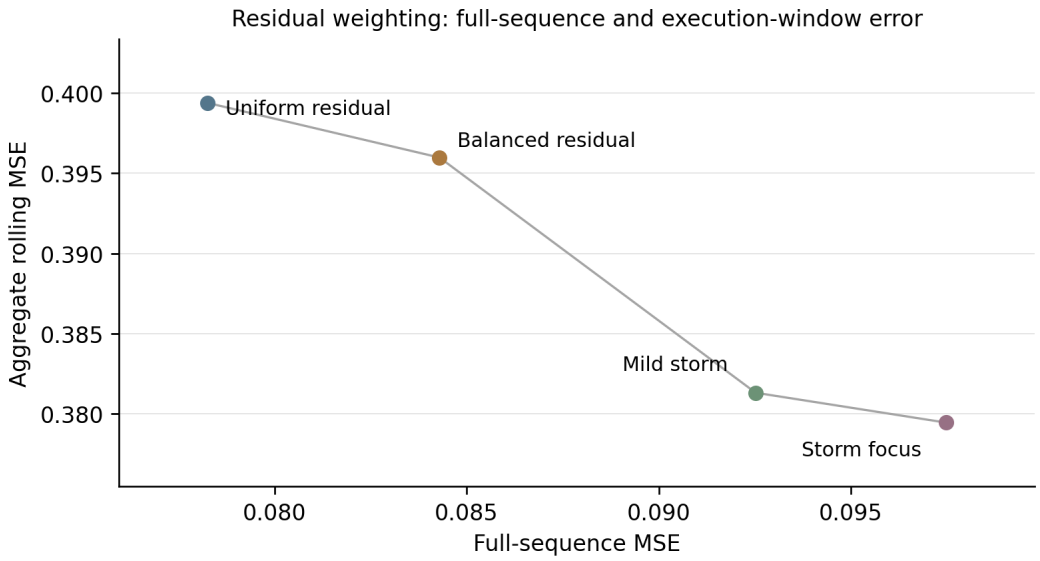

Figure 6: Residual-sweep tradeoff between full-sequence MSE and aggregate rolling MSE. Each point is a configuration from the lower panel of Table 1.

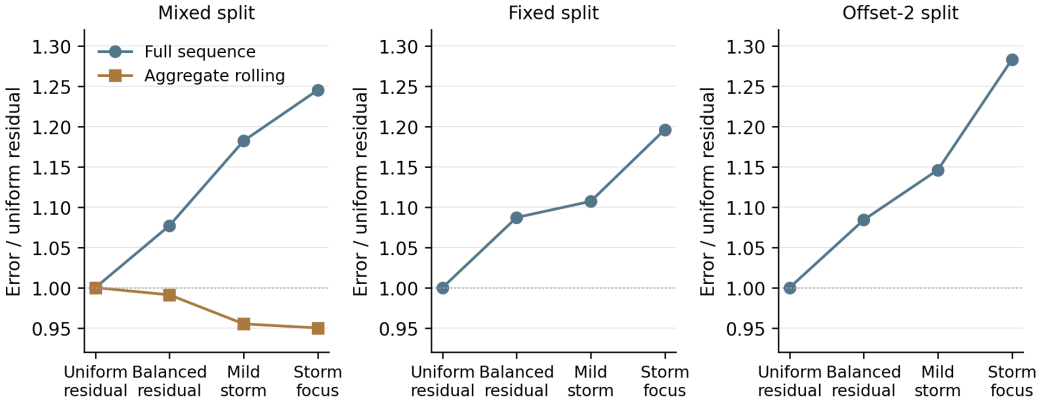

Figure 7: Residual-sweep error ratios across the mixed, fixed, and offset-2 protocols, using the values in Table 13. Aggregate mixed rolling error decreases as full-sequence and forward policy errors increase.

tion without satisfying the success event. The reported effects are therefore matched differences at a converged common budget; they are not a stronger-backbone or state-of-the-art Push-T comparison.

D.7 Difficulty-Stratified Closed-Loop Evaluation

We partition each model's 200 evaluation episodes into quartiles by initial task difficulty: the normalized T-block center-distance to its goal plus normalized wrapped angle difference. Seeded resets give the same initial conditions across arms. Uniform success decreases from 0.256 in Q0 to 0.136 in Q3 (Table 17; Figure 16). In Q3, uniform exceeds the calibrated residual and calibrated Meta-Weight-Net models on success (0.136 versus 0.116 and 0.072) and coverage (0.757 versus 0.715 and 0.673). No-align has a slightly higher Q3 success mean, 0.140, but lower coverage, 0.732. Both calibrated Meta-Weight-Net and L2RW have lower success and coverage means than uniform in every quartile.

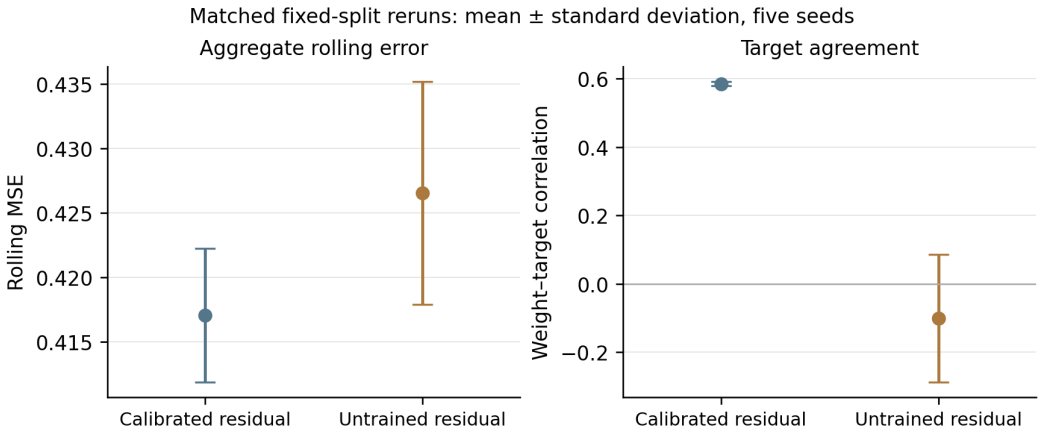

Figure 8: Fixed-split matched reruns. Points show means and error bars show sample standard deviations over five seeds for aggregate rolling MSE and weight–target correlation (Table 2).

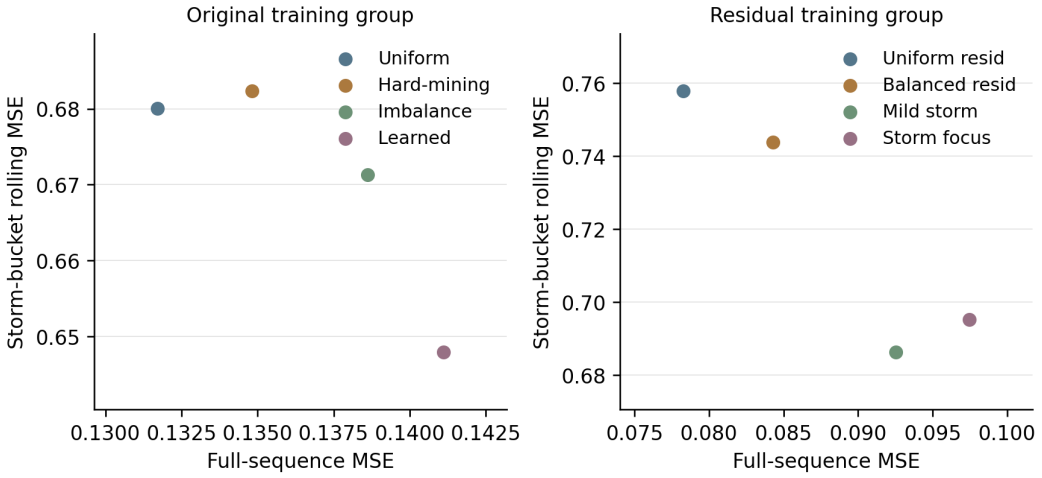

Figure 9: Full-sequence error and storm-bucket rolling error shown separately for the original and residual training groups. The two groups use different recipes. Within the residual sweep, mild storm has lower storm error than storm focus.

D.8 Noise-Injection Stress Test

Protocol. We reuse the Push-T corpus, backbone, episode splits, and importance-network recipe of Appendix D.6: an 8-D feature vector, softplus weights clamped to $[0.25, 2]$ before batch-mean normalization, and $8{,}000$ AdamW steps at learning rate 3×10^{-4} and batch size 256. We add Gaussian noise to target-position offsets in 20% of training episodes, with standard deviation 12 px in the 0–512 coordinate frame. Meta and validation pools, and the closed-loop environment, remain clean. Each arm uses seeds 11, 22, 33, 44, and 55, with 200 evaluation episodes per trained model.

The added `qalign` arm receives a label-consistency feature and uses a calibration target consisting of task intensity minus an inconsistency penalty. Severity-only targets instead assign greater priority to the increased path length and jerk induced by corruption. We measure this preference by the mean weight on corrupted windows divided by the mean weight on clean windows. A ratio below 1 indicates lower weight on corrupted data. Weight–noise correlation measures agreement with the binary corruption indicator.

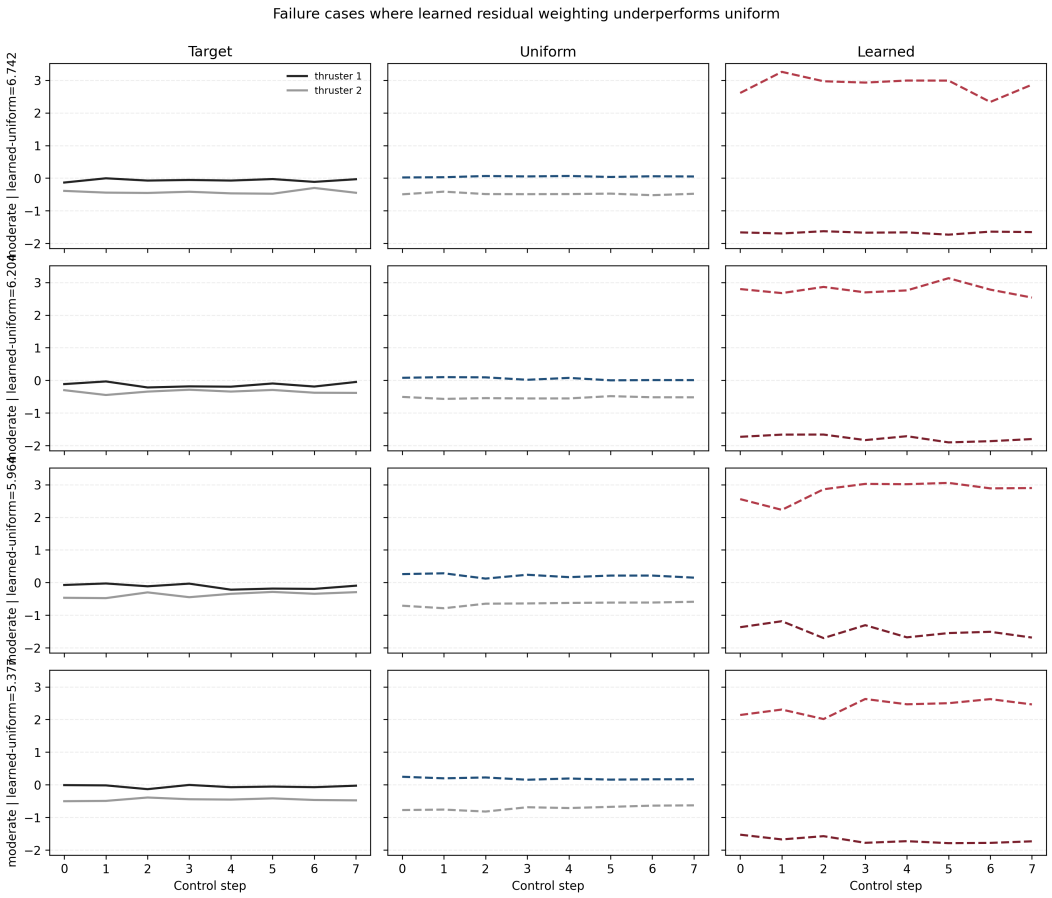

Figure 10: Representative fixed-split failure cases where aligned learned weighting underperforms uniform. Each row shows the target action sequence together with the uniform and learned predictions on windows with the largest learned-minus-uniform error gaps.

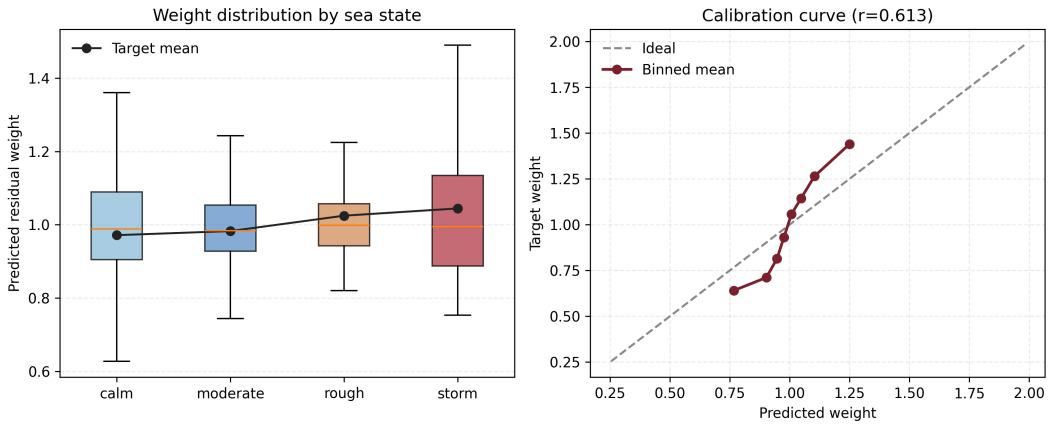

Figure 11: Sea-state calibration of the learned residual weights. Predicted mean weights closely track the calibration-target ordering from calm to storm.

Consistency feature and consistency-aware target. The label-consistency signal is computed from each window's own action label and observed state, without access to the corruption indicator. Let a_0 be the agent position at the observation time, $\mathrm{off}_i = A_{i,0} - a_0$ the offset of the first com-

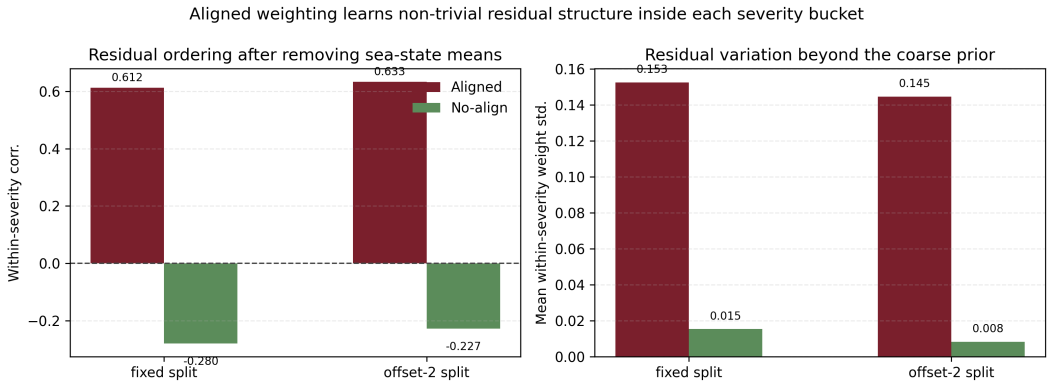

Figure 12: Within-severity diagnostics for the forward-diagnostic pair. After subtracting each bucket mean, calibrated weights retain positive target correlation and greater dispersion than the untrained residual control.

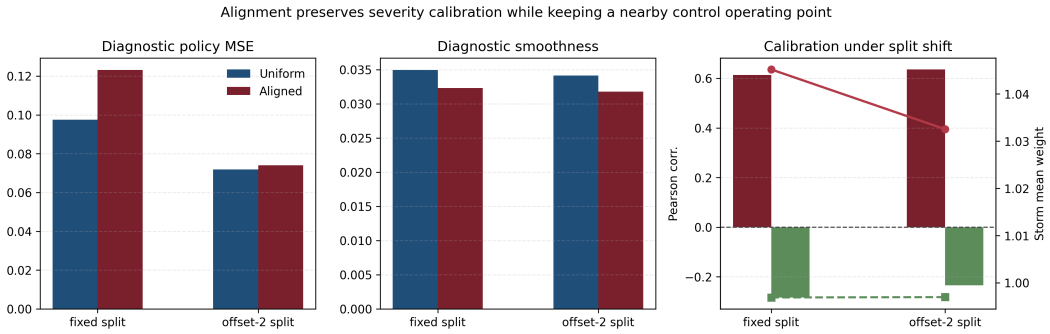

Figure 13: Forward diagnostics on fixed and offset-2 splits. The aligned model retains positive target correlation across the shift, while the no-align control has negative correlation. Task errors and calibration are reported separately in Table 12.

manded target from the agent, and $\mathrm{disp}_i = a_0 - a_{-\mathrm{HIST}+1}$ the agent's motion across the observation history. The consistency is their cosine alignment,

$$c_i = \frac{\mathrm{off}_i \cdot \mathrm{disp}_i}{\|\mathrm{off}_i\| \, \|\mathrm{disp}_i\| + \varepsilon}.$$

On clean expert windows the agent tends to move toward its commanded target, giving a batch-mean consistency of $c_i \approx 0.91$; the Gaussian offset corruption used here scrambles this alignment, lowering the batch mean over corrupted windows to ≈ 0.63 (the mean over corrupted windows, not a single fully-randomized window, which would fall near zero; other corruption modes reach 0.48–0.84, Table 19). When used as an importance-network input, c_i is standardized across the batch. The consistency-aware target subtracts a standardized inconsistency penalty from the severity target of Appendix B.2:

$$\mathrm{incon}_i = 1 - c_i, \qquad q_i = y_i^{\mathrm{sev}} - \lambda_{\mathrm{cons}} \widetilde{\mathrm{incon}}_i, \qquad \lambda_{\mathrm{cons}} = 1.0,$$

where $\widetilde{\mathrm{incon}}_i$ is batch-standardized and q_i is then shifted to be positive and renormalized to unit batch mean, exactly as for the severity target. The Push-T severity target itself is built from four label-derived task-intensity components—future block-centroid displacement, action path length $\sum_t |A_{i,t+1} - A_{i,t}|$, action jerk $\mathrm{mean}_t (A_{i,t+2} - 2A_{i,t+1} + A_{i,t})^2$, and agent-to-block distance—together with the detached per-sample loss.

Feature $\times$ target ablation and the direct-target control. To separate the effect of the consistency *feature* from that of the consistency-aware *target*, we run the full 2×2 grid: the importance-network input is either the base feature vector or the base vector with the consistency channel

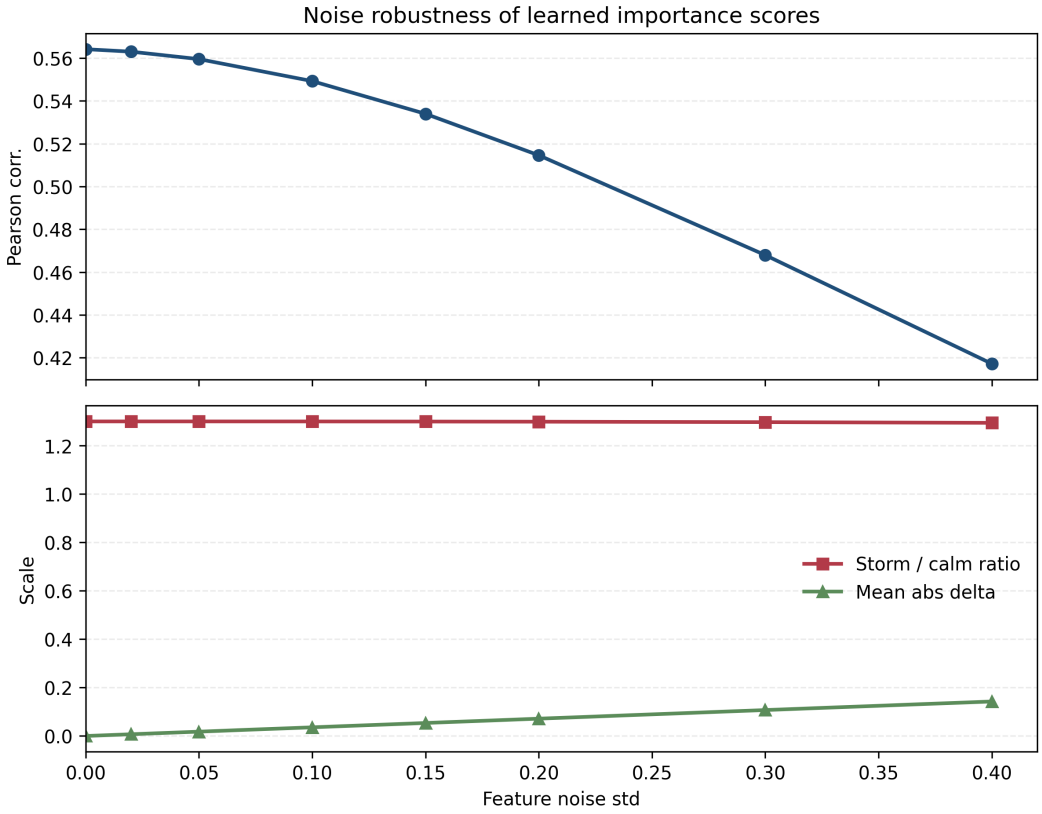

Figure 14: Noise robustness of the learned importance scores. Alignment degrades smoothly, but storm emphasis stays nearly flat.

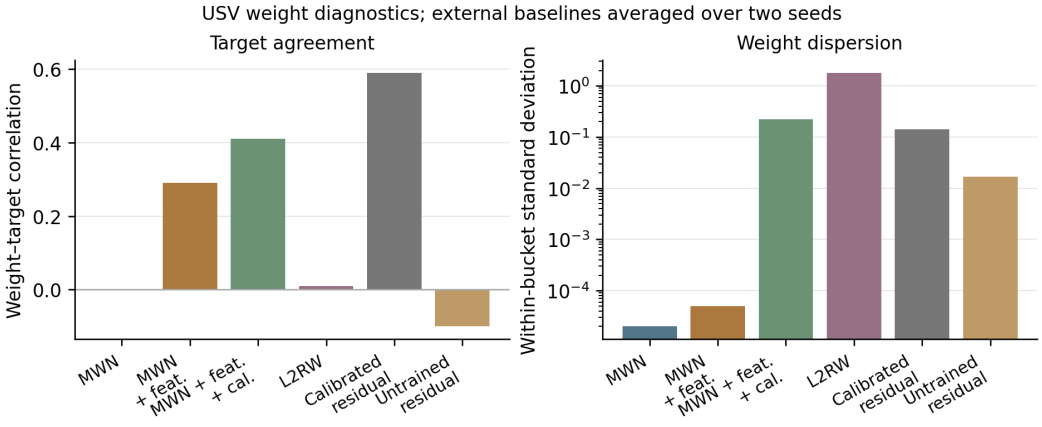

Figure 15: External USV weight diagnostics, with means from Table 3. Feature-conditioned Meta-Weight-Net has positive correlation but near-zero dispersion. Calibration increases the dispersion and correlation, whereas L2RW assigns dispersed weights with low severity correlation.

appended, and the calibration target is either severity-only or consistency-aware. The four cells are `aligned` (base, severity), `align_cfeat` (base+consistency, severity), `qalign_basefeat` (base, consistency-aware), and `qalign` (base+consistency, consistency-aware). We additionally include a direct-target control that removes the importance network entirely and uses the detached calibration target as the sample weight: `direct` for the severity target and `direct_qual` for the

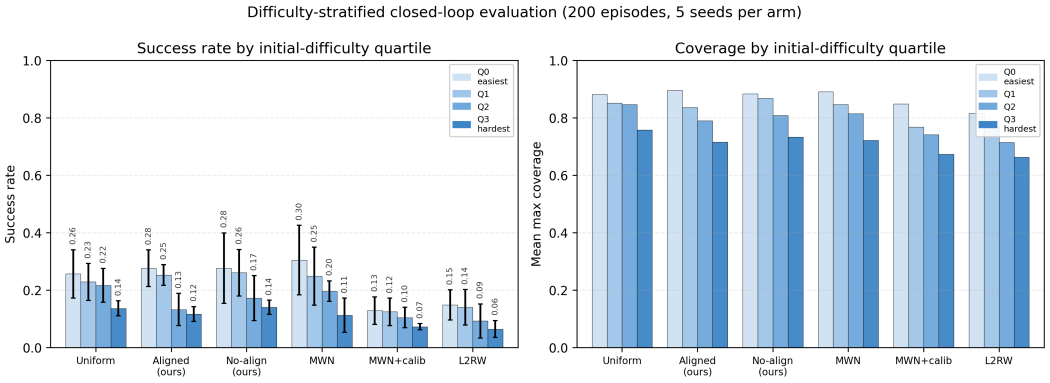

Figure 16: Closed-loop outcomes by initial-difficulty quartile. Uniform success decreases with initial difficulty. Calibrated residual weighting and calibrated Meta-Weight-Net have lower Q3 success and coverage than uniform; no-align has the highest Q3 success mean.

consistency-aware target. These controls test whether learning a mapping from observable features contributes anything beyond the target it is trained to fit.

Exploratory corruption boundary: is the preference component-specific? A single noise mode cannot tell whether the severity target's preference is specific to the components a corruption inflates. We therefore repeat the direct-target study under four synthetic modes that dissociate action path length and jerk from label consistency, holding the noise fraction at 20% and using three seeds per new mode (the Gaussian mode reuses the five-seed run above). This is a mode comparison at a fixed prevalence, not a fraction-or-magnitude robustness sweep. *Gaussian* and *high-frequency* noise both inflate jerk (the latter by $66\times$ relative to clean windows); a *constant offset* leaves path length and jerk essentially unchanged while reducing consistency; a *time shift* mildly raises jerk. Table 19 reports the resulting component and weight ratios together with closed-loop success.

Across the tested modes, the observed ordering follows the severity components: the `direct` noisy-to-clean ratio is 1.89 (Gaussian), 2.11 (high-frequency), and 1.17 (time shift), but 1.03 for the constant offset, which leaves jerk and path length nearly unchanged. The consistency-aware target gives ratios near one in the jerk-inflating modes and 0.81 under the constant offset. Under time shift its success mean is 0.173 versus uniform 0.155, but three seeds are insufficient for a reliable advantage claim. We therefore use this study to generate and check the component-level explanation, not to claim universal corruption robustness. The deterministic leave-one-component-out calculation below further localizes the observed weight preference to jerk and path length.

Component attribution: which target term drives the preference? The boundary study establishes an association—the up-weighting co-occurs with inflated jerk and path length—but not which term of the severity target produces it. Because the `direct` weight is the detached severity target, its noisy/clean ratio is a deterministic function of the label-derived components, so we recompute it with each component removed, without retraining (Table 20; three seeds). Two results sharpen the attribution. First, the preference is carried almost entirely by action jerk and path length: dropping jerk lowers the Gaussian ratio from 1.91 to 1.46 and dropping path length to 1.64, while a jerk-only target reaches 7.2 (Gaussian) and 9.4 (high-frequency); the block-displacement and agent-to-block components are individually neutral (≈ 1.00 and 0.99). Second, the detached per-sample loss term is not the cause: the four-component target that *excludes* the loss gives essentially the trained `direct` ratio (1.91 vs. 1.89 Gaussian, 2.23 vs. 2.11 high-frequency, 0.99 vs. 1.03 constant-offset), so the constant offset's near-neutrality is not an artifact of the loss term absorbing it. This lets us state the mechanism as an attribution to the dynamical-roughness components (jerk, path length) rather than only an association, and confirms that no label-derived component detects the constant-offset corruption. Figure 18 visualizes both the boundary ratios and this component attribution.

Hard-but-correct negative control: does the correction read reliability or smoothness? The consistency-aware target down-weights corrupted labels, but the paper's motivation also requires

that it *preserve* genuinely hard but correct maneuvers rather than suppress all non-smooth actions. We test this with a matched control on clean expert data (no injected noise, five seeds). We take the top quartile of clean training windows by action jerk as genuine hard maneuvers (*clean-hard*; sharp turns and contact adjustments) and the remaining windows as *clean-easy*. We then form a matched corrupted twin (*flip*) by reflecting each clean-hard action chunk about the agent's current position, $A' = 2a_0 - A$. Reflection negates every temporal difference of the action, so action jerk and path length are preserved *exactly*, while the first commanded offset is inverted and label consistency flips sign. The flip twin is therefore a jerk- and path-matched corruption that differs from clean-hard only in label reliability. Weights are the target values themselves, normalized to overall mean one over the pool, with the detached loss channel omitted (shown immaterial by the leave-one-out attribution above). Because the two hard groups share identical severity components, the severity target assigns them equal weight by construction (1.21 each), so it cannot separate a valid hard maneuver from a matched corruption. The consistency-aware target instead keeps clean-hard up-weighted (1.19, above clean-easy 1.12) while collapsing the flip twin to 0.45, a preservation gap of $+0.74$ (Table 21). Clean-hard maneuvers also retain high natural consistency (0.82, versus 0.94 for clean-easy and -0.82 for the flip twin), so they are not mistaken for corruption. The correction thus responds to label reliability, not to dynamical roughness: it retains valid high-intensity control and removes only the matched corruption.

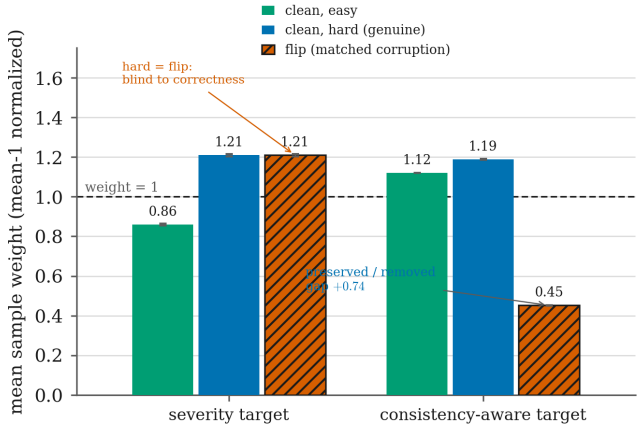

Figure 17: Hard-but-correct negative control (clean Push-T expert data, five seeds). Bars are mean sample weights (overall mean one) for three groups under the severity target (left) and the consistency-aware target (right). The *flip* twin has action jerk and path length identical to the genuine hard maneuvers by construction, differing only in inverted label consistency. Under the severity target, clean-hard and flip receive the same weight (1.21), so it cannot separate a valid hard maneuver from a matched corruption. Under the consistency-aware target, clean-hard stays up-weighted (1.19) while the matched flip twin collapses to 0.45: the correction preserves valid high-intensity control and removes only the corruption. Error bars are sample standard deviations over seeds. Numbers are in Table 21.

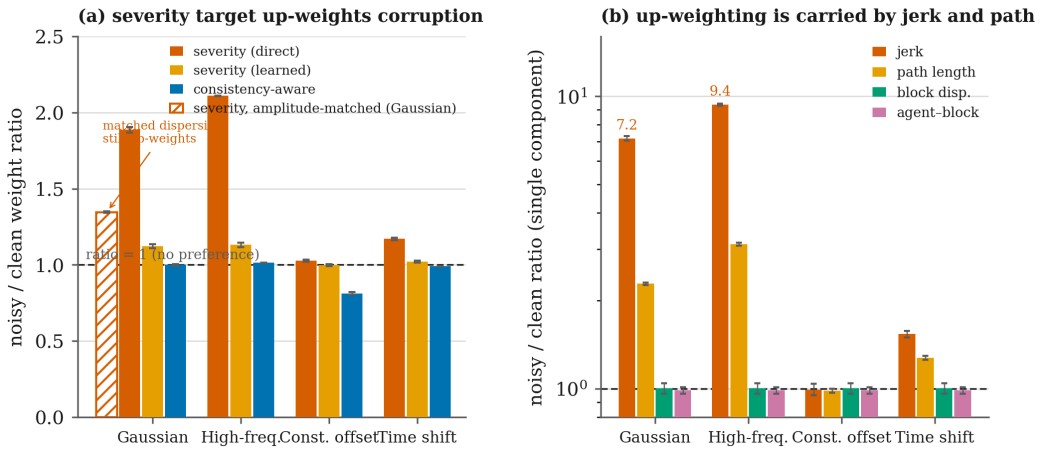

Figure 18: Corruption-boundary study on Push-T (20% corrupted training episodes). **(a)** Noisy/clean sample-weight ratio per corruption mode for the severity target applied directly (`direct`), through the learned network (`aligned`), and the consistency-aware target applied directly (`direct_qual`); bars above the dashed line up-weight corrupted windows. The severity target up-weights every jerk-inflating mode but is near-neutral on the constant offset, which leaves the severity components unchanged; the consistency-aware target is neutral-to-suppressive throughout. The hatched bar is the amplitude-matched control (`direct_shrink`, Gaussian only): its within-bucket dispersion matches `direct_qual` yet it still up-weights corrupted windows (ratio 1.35), so matching amplitude does not remove the preference. **(b)** Leave-one-component attribution (log scale): the noisy/clean ratio of each severity component used alone as the target. Action jerk (annotated) and path length carry the up-weighting, while block-displacement and agent-to-block are individually neutral, and none is sensitive to the constant offset. Error bars are sample standard deviations over seeds (Gaussian n=5 in (a), n=3 elsewhere; n=3 throughout (b)). Numbers are in Tables 19 and 20.

Table 5: Push-T with noise in 20% of training episodes; five-seed means and sample standard deviations, with clean closed-loop evaluation. The consistency-aware target assigns the lowest noisy-to-clean weight ratio. Severity-only calibrated models up-weight corrupted windows. The `direct` arms use the calibration target as the sample weight with no importance network, so their weight–target correlation is one by construction. `direct_shrink` is an amplitude-matched control, $w = 1+\alpha(w^{\text{sev}}-1)$ with $\alpha = 0.43$, that preserves the severity ordering but scales the deviation from one so its within-bucket dispersion matches `direct_qual`; it still up-weights corrupted windows (ratio 1.35) and its success stays at the severity level. `align_cfeat` (base+consistency feature, severity target) and `qalign_basefeat` (base feature, consistency-aware target) are the off-diagonal cells of the feature × target ablation whose diagonal cells are `aligned` and `qalign`. L2RW ratios are omitted because its per-batch meta-gradient weights are not evaluated on the held-out diagnostic split.

Arm	Success	Coverage	Val MSE	W–target corr	Within-bucket std	Noisy/clean ratio	W–noise corr
`uniform`	0.191 ± 0.071	0.817	0.0649	—	0.000 ± 0.000	1.000 ± 0.000	—
`aligned`	0.133 ± 0.017	0.795	0.0657	$+0.504 \pm 0.034$	0.204 ± 0.022	1.122 ± 0.014	$+0.178$
`qalign`	0.181 ± 0.024	0.806	0.0648	$+0.907 \pm 0.006$	0.086 ± 0.002	$\mathbf{0.913 \pm 0.004}$	-0.214
`noalign`	0.177 ± 0.058	0.818	0.0650	-0.024 ± 0.267	0.030 ± 0.006	0.997 ± 0.004	-0.039
`mwn`	0.189 ± 0.031	0.812	0.0653	$+0.355 \pm 0.502$	0.000 ± 0.000	1.000 ± 0.000	$+0.095$
`mwn_align`	0.102 ± 0.016	0.759	0.0657	$+0.725 \pm 0.016$	0.241 ± 0.003	1.228 ± 0.011	$+0.237$
`l2rw`	0.096 ± 0.035	0.739	0.0721	$+0.185 \pm 0.059$	2.338 ± 0.320	—	—
`direct`	0.114 ± 0.015	0.756	0.0674	$+1.000 \pm 0.000$	0.187 ± 0.007	1.887 ± 0.018	$+0.643$
`direct_shrink`	0.133 ± 0.036	0.777	0.0658	$+1.000 \pm 0.000$	0.081 ± 0.003	1.346 ± 0.004	$+0.640$
`direct_qual`	0.184 ± 0.053	0.808	0.0649	$+1.000 \pm 0.000$	0.081 ± 0.003	1.000 ± 0.004	$+0.000$
`align_cfeat`	0.157 ± 0.049	0.800	0.0664	$+0.518 \pm 0.024$	0.227 ± 0.013	1.312 ± 0.012	$+0.375$
`qalign_basefeat`	0.167 ± 0.052	0.810	0.0648	$+0.392 \pm 0.024$	0.073 ± 0.006	1.000 ± 0.002	-0.002

Table 6: Fixed importance-module hyperparameters.

Parameter	Value
Hidden width	128
Temperature T	1.0
Weight floor $w_{\min}$	0.25
Weight ceiling $w_{\max}$	4.0
Alignment weight λ_{align}	0.5
Wave difficulty α_{wave}	0.35
Policy difficulty α_{policy}	0.5
Sea-state priors $(\beta_{\text{calm}}, \beta_{\text{moderate}}, \beta_{\text{rough}}, \beta_{\text{storm}})$	$(1.0, 1.1, 1.3, 1.6)$

Table 7: Run glossary. Names in the left column are the display labels used in the result tables and figures; runs in different groups use different training configurations and are not directly comparable.

Display name(s) in tables/figures	Canonical run	Group
Uniform baseline / Uniform	`uniform`	original mixed-split recipe
Hard-mining baseline / Hard-mining	`hard_mining`	original mixed-split recipe
Imbalance baseline / Imbalance	`imbalance`	original mixed-split recipe
Learned weighting / Learned	`learned`	original mixed-split recipe
Uniform resid	`uniform_resid`	residual-weighting family
Balanced resid	`balanced_resid`	residual-weighting family
Mild storm	`mild_storm`	residual-weighting family
Storm focus	`storm_focus`	residual-weighting family
Prior-only	`prior_only`	residual-weighting family
noalign (matched)	`noalign_matched`	residual-weighting family, matched to `uniform_resid`
Aligned	`aligned`	forward-diagnostic run (fixed / offset-2 protocol)
No-align	`noalign`	forward-diagnostic run, matched to `aligned`

Table 8: The 17-D feature vector used by the importance network.

Block	Components
Sea state (4)	calm, moderate, rough, storm one-hot
Goal (3)	$\lvert g^{\text{heading}}\rvert/\pi$, $\min(\lvert g^{\text{speed}}\rvert, 20)/20$, $\min(\lvert g^{\text{dist}}\rvert, 50)/50$
Wave history (4)	energy, mean abs, std, absolute trend
State history (4)	mean speed, speed std, mean abs yaw rate, mean abs jerk
Wave prediction (2)	predicted energy, predicted std

Table 9: Baseline taxonomy used in the paper.

Setting	Instance(s)	Role in the paper
Optimization reference	uniform	no weighting, no learned net
Heuristic-only weighting	imbalance / hard-mining	non-learned severity or difficulty emphasis
Aligned learned model	learned	original residual calibrator under the mixed-split recipe
Uniform sea-state control	uniform_resid	learned weighting with a uniform coarse prior
Frozen coarse-prior control	prior_only	same training skeleton, residual multiplier fixed to 1
Residual operating sweep	balanced_resid / mild_storm / storm_focus	calibrated tradeoff family inside learned weighting
Calibration ablation	noalign	untrained residual net; alignment removed

Table 10: Prior-only and calibrated residual configurations. The prior-only multiplier is fixed to 1. The calibrated configuration has lower errors in each displayed comparison; its sea-state coefficients are uniform while its energy factor remains active.

Method	Mixed wave	Mixed full-seq	Mixed rolling MSE	Fixed policy	Offset-2 policy
prior_only	0.869278	0.084686	0.405440	0.109551	0.069853
uniform_resid	0.800950	0.078265	0.399369	0.100912	0.066796

Table 11: Within-severity residual structure after subtracting each sea-state bucket's mean weight.

Split	Method	Within-severity corr.	Mean within-severity weight std.
fixed	Aligned	0.611896	0.152617
fixed	No-align	-0.279658	0.015317
offset-2	Aligned	0.633467	0.144593
offset-2	No-align	-0.227352	0.008303

Table 12: Forward diagnostics on the fixed split and the held-out offset-2 shift.

Split	Method	Wave MSE	Policy MSE	Smoothness	Weight-target corr.
fixed	Uniform	0.600601	0.097546	0.034960	n/a
fixed	Aligned	0.840921	0.123132	0.032307	0.613128
fixed	No-align	0.585563	0.080826	0.030320	-0.283648
offset-2	Uniform	0.356815	0.071821	0.034136	n/a
offset-2	Aligned	0.580676	0.074053	0.031783	0.635887
offset-2	No-align	0.352145	0.058640	0.029935	-0.234536

Table 13: Residual-sweep metrics normalized by `uniform_resid` within each evaluation protocol. The mixed rolling column is aggregated across sea states. Stronger storm emphasis reduces this aggregate metric while increasing full-sequence and forward policy errors. Table 15 reports storm-bucket errors separately.

Method	Mixed full-seq / U.R.	Mixed rolling MSE / U.R.	Fixed policy / U.R.	Offset-2 policy / U.R.
uniform_resid	1.000	1.000	1.000	1.000
balanced_resid	1.077	0.991	1.087	1.084
mild_storm	1.182	0.955	1.107	1.146
storm_focus	1.245	0.950	1.196	1.283

Table 14: Storm-bucket rolling MSE within the original training group. Learned weighting has the lowest error among these four configurations.

Run	Storm rolling MSE
Uniform baseline	0.680057
Hard-mining baseline	0.682323
Imbalance baseline	0.671307
Learned weighting	0.647966

Table 15: Sea-state policy rolling MSE on the mixed split. Different weighting rules perform best in different severity buckets.

Method	Calm	Moderate	Rough	Storm
Uniform	0.416288	0.282155	0.220246	0.680057
Hard-mining	0.416040	0.283169	0.220875	0.682323
Imbalance	0.435879	0.294293	0.226047	0.671307
Learned	0.434847	0.302187	0.228501	0.647966
Prior-only	0.425707	0.276170	0.210431	0.806674
Uniform resid	0.425302	0.273731	0.227696	0.757886
Balanced resid	0.428238	0.274436	0.216729	0.743943
Mild storm	0.415797	0.276024	0.211870	0.686329
Storm focus	0.408269	0.270123	0.216055	0.695319
No-align matched	0.438871	0.283830	0.213936	0.853078

Table 16: Learned weights keep nearly the same storm emphasis under stronger feature noise.

Noise std	Noisy-target corr	Base-noisy corr	Storm/Calm ratio
0.00	0.564142	1.000000	1.300536
0.02	0.562992	0.999160	1.300579
0.05	0.559509	0.994780	1.300586
0.10	0.549235	0.979499	1.300430
0.15	0.533947	0.955221	1.300068
0.20	0.514584	0.923557	1.299461
0.30	0.467925	0.845294	1.297570
0.40	0.417170	0.758506	1.294699

Table 17: Closed-loop success and coverage by initial-difficulty quartile, averaged over five training seeds and 200 evaluation episodes per model. No-align has the highest Q3 success mean; uniform has the highest Q3 coverage. Both calibrated arms are below uniform on the two Q3 metrics.

	Success				Coverage			
Arm	Q0	Q1	Q2	Q3	Q0	Q1	Q2	Q3
`uniform`	0.256	0.228	0.216	0.136	0.882	0.851	0.846	**0.757**
`aligned`	0.276	0.252	0.132	0.116	0.896	0.836	0.790	0.715
`noalign`	0.276	0.260	0.172	**0.140**	0.883	0.868	0.808	0.732
`mwn`	0.304	0.248	0.196	0.112	0.890	0.846	0.814	0.721
`mwn_align`	0.128	0.124	0.104	0.072	0.848	0.768	0.741	0.673
`l2rw`	0.148	0.140	0.092	0.064	0.815	0.771	0.714	0.662

Table 18: Paired-seed success differences relative to uniform. Two-sided p-values are recomputed from the displayed rounded t statistics with four degrees of freedom (five seed pairs); values are approximate. At the unadjusted 0.05 threshold, the calibrated Meta-Weight-Net comparison is significant.

Arm	Δ success vs uniform	t	p (two-sided)
`aligned`	-0.058	-1.80	0.146
`qalign`	-0.010	-0.40	0.710
`noalign`	-0.014	-0.51	0.637
`mwn`	-0.002	-0.06	0.955
`mwn_align`	-0.089	-3.07	$\mathbf{0.037}$
`l2rw`	-0.095	-2.62	0.059

Table 19: Exploratory corruption-mode comparison on Push-T at a fixed 20% corruption fraction (Gaussian n=5, other modes n=3; means $\pm$ sample standard deviations). "jerk" and "path" are noisy/clean component ratios; "cons." is label consistency on clean versus noisy windows. The results support a component-level mechanism but do not constitute a fraction-or-magnitude robustness sweep.

Corruption	jerk	path	cons. clean→noisy	`direct` ratio	`aligned` ratio	`direct_qual` ratio	verdict
Gaussian	—	—	$0.91 \to 0.63$	1.887 ± 0.018	1.122 ± 0.014	1.000 ± 0.004	severity up-weights
High-frequency	65.9 ± 1.0	3.76 ± 0.05	$0.91 \to 0.59$	2.110 ± 0.002	1.131 ± 0.016	1.013 ± 0.003	severity up-weights
Constant offset	0.99 ± 0.09	0.98 ± 0.02	$0.91 \to 0.48$	$\mathbf{1.028 \pm 0.007}$	0.999 ± 0.007	0.813 ± 0.008	severity near-neutral
Time shift	1.94 ± 0.10	1.32 ± 0.03	$0.91 \to 0.84$	1.170 ± 0.007	1.021 ± 0.006	0.990 ± 0.002	severity up-weights

Corruption	`uniform`	`direct`	`aligned`	`direct_qual`
Gaussian	0.191 ± 0.071	0.114 ± 0.015	0.133 ± 0.017	0.184 ± 0.053
High-frequency	0.247 ± 0.045	0.128 ± 0.039	0.182 ± 0.008	0.197 ± 0.008
Constant offset	0.122 ± 0.052	0.083 ± 0.038	0.057 ± 0.021	0.113 ± 0.029
Time shift	0.155 ± 0.030	0.070 ± 0.018	0.117 ± 0.010	0.173 ± 0.041

Closed-loop success by corruption mode and arm (means $\pm$ sample standard deviations).

Table 20: Exploratory leave-one-component-out attribution for the direct severity target (label-derived components only). Entries are noisy/clean ratios with a component removed ("drop") or used alone ("only"), over three seeds at corruption fraction 0.2. In these tested modes, action jerk and path length carry the up-weighting; the other components are individually near-neutral.

	drop one component (keep the other three)					single-component target			
Corruption	all 4	−block_disp	−path	−jerk	−agent_block	block_disp	path	jerk	agent_block
Gaussian	1.91	2.12	1.64	1.46	2.37	1.00	2.29	7.18	0.99
High-frequency	2.23	2.56	1.80	1.63	2.96	1.00	3.12	9.36	0.99
Constant offset	0.99	0.99	0.99	0.99	0.99	1.00	0.99	0.99	0.99
Time shift	1.16	1.19	1.09	1.11	1.24	1.00	1.28	1.54	0.99

Table 21: Hard-but-correct negative control on clean Push-T expert data (five seeds; means ± sample standard deviations). *clean-hard* are the top-quartile clean windows by action jerk; *flip* reflects their action chunk about the agent position, preserving jerk and path length exactly while inverting label consistency. Weights are target values normalized to overall mean one (loss channel omitted). The severity target gives the two jerk-matched groups identical weight; the consistency-aware target preserves the genuine hard maneuvers and collapses only the matched corruption.

Group	act. jerk	consistency (cos)	severity weight	consistency-aware weight
clean-easy	9.1 ± 0.1	0.941 ± 0.001	0.860 ± 0.004	1.120 ± 0.001
clean-hard (genuine)	116.1 ± 2.8	0.823 ± 0.002	1.210 ± 0.006	$\mathbf{1.187 \pm 0.002}$
flip (matched)	116.1 ± 2.8	-0.823 ± 0.002	1.210 ± 0.006	$\mathbf{0.452 \pm 0.001}$