Test-Time Prompt-Agnostic Decomposition

Junze Wang[1] **Lei Fan**[2] **Dezheng Zhang**[1] **Donglin Di**[3] **Yang Song**[2] **Sidong Liu**[4] **Cong Cong**[4*]
[1]University of Science and Technology Beijing [2]University of New South Wales
[3]Sympflux Co., Ltd. [4]Macquarie University
wangjunze@yeah.net,thomas.cong@mq.edu.au

Abstract

Test-Time Prompt Tuning (TPT) adapts large pretrained models to unseen target distribution shifts by updating only lightweight prompt tokens. Existing TPT methods are effective, but they mainly optimize each prompt update locally. Across test-time steps, prompts updated at the current step are reused for the corresponding predictions and updates, so noisy or biased unlabeled objectives can accumulate and destabilize the prompt-update trajectory. We identify two failure modes: magnitude expansion, where updates move the prompt too far, and directional drift, where consecutive updates point in inconsistent directions. To analyze and control these failures, we propose Test-Time Prompt-Agnostic Decomposition (TPD), which characterizes observed prompt-update trajectories from a dynamical-systems view and decomposes base prompt updates into magnitude and direction. TPD computes a spectral radius to measure update expansion and decomposes update directions into persistent, oscillatory, and residual components. Building on this decomposition, Adaptive Koopman Control (AKC) regulates update magnitude by shrinking updates under expansive recent dynamics, while Hankel Update Router (HUR) refines update direction by preserving persistent components and suppressing oscillatory and residual components. Together, AKC and HUR produce a stabilized prompt update. As a prompt-agnostic framework, TPD can be plugged into visual and text TPT methods without modifying the backbone, prompt architecture, or adaptation loss. Experiments on 15 datasets show that TPD consistently improves 10 baselines, with accuracy gains of 3–8% for visual prompts and over 2% for text prompts[1].

1 Introduction

Test-Time Prompt Tuning (TPT) has emerged as a parameter-efficient paradigm for adapting large pretrained vision and vision-language models, such as ViT [Dosovitskiy et al., 2020] and CLIP [Radford et al., 2021], to unseen target distribution shifts [Yuan et al., 2023]. Instead of updating the full backbone, TPT adapts only prompt tokens during inference, making it well suited for frozen foundation models [Liang et al., 2025]. Existing methods adapt prompts from unlabeled test data using step-level objectives, including entropy minimization [Shu et al., 2022], confidence-based augmentation selection [Feng et al., 2023, Farina et al., 2024], prompt calibration [Yoon et al., 2024], prompt-history regularization [Ma et al., 2023, Xiao et al., 2025], and black-box prompt updates [Meng et al., 2025]. Although effective, these methods primarily optimize each prompt update locally. This local-update view is incomplete for TPT across steps: prompts updated at the current step are used for the corresponding predictions and updates, so noisy or biased updates from unlabeled objectives can influence later prompt states and accumulate over time [Niu et al., 2023].

This cross-step dependency exposes a key weakness of existing TPT methods. Consecutive updates may move with inappropriate magnitudes or point in inconsistent directions, leading to prompt-update trajectory instability. As illustrated in Figure 1(a), CLIP-based text TPT shows pronounced

*Corresponding author.
[1]The code is available at https://github.com/reeive/TPD.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

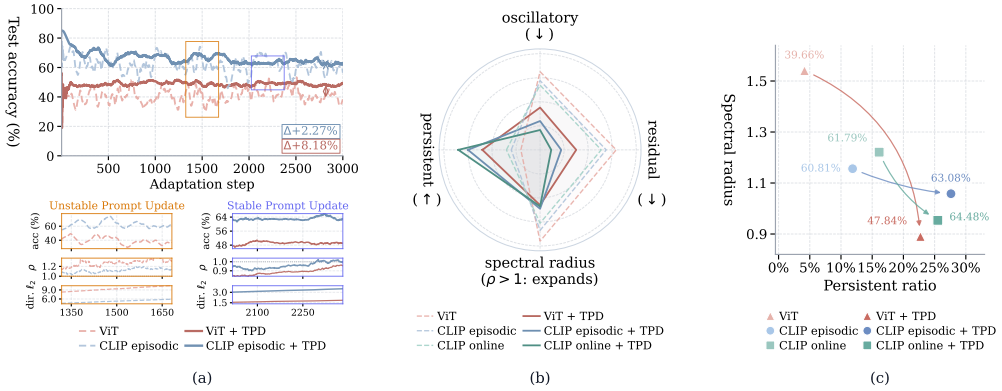


Figure 1: (a) The accuracy of ViT visual and CLIP text prompts in episodic modes, averaged over four ImageNet variants. Gains denote TPD improvements. The zoomed windows show accuracy, magnitude (spectral radius ρ), and direction (ℓ_2 distance between consecutive update vectors, decomposed in (b)) in unstable and stable regimes. (b) TPD decomposes each prompt update into persistent, oscillatory, and residual components, with the spectral radius measuring local expansion. (c) Stability-performance relation summarized by spectral radius and persistent ratio.

oscillations in accuracy and update behavior, while ViT-based visual TPT exhibits unstable update behavior together with persistently low accuracy across test steps. These observations indicate that existing methods lack a principled tool for analyzing and controlling prompt updates across test steps. To better characterize prompt-update trajectories, we decompose each update into two complementary components: magnitude, which describes the update step size, and direction, which describes the update orientation [Bottou et al., 2018]. This magnitude-direction view provides a structured way to analyze prompt-update trajectories: instability may arise when update magnitudes expand across steps, when update directions drift or oscillate over time, or when both occur jointly.

Accordingly, we propose Test-Time Prompt-Agnostic Decomposition (TPD), which characterizes observed prompt-update trajectories from a dynamical-systems perspective [Williams et al., 2015]. TPD operationalizes the magnitude–direction view through the prompt-update decomposition, enabling both analysis and stabilization across test steps. As illustrated in Figure 1(b), TPD provides a quantitative interpretation of the unstable behaviors observed in Figure 1(a) by decomposing trajectory dynamics into two complementary branches. For the magnitude branch, TPD computes a spectral radius as an expansion measure, where larger values indicate stronger growth of update magnitudes across recent steps. For the direction branch, TPD decomposes each update into three additive components: a persistent component capturing consistent adaptation signals, an oscillatory component capturing sign-flipping interference, and a residual component capturing unexplained noise. These quantities provide an explicit basis for magnitude expansion and directional drift.

The decomposition quantities reveal a stability pattern in Figure 1(c): methods with a large spectral radius and a low persistent ratio consistently exhibit poor performance. Specifically, the spectral radius summarizes magnitude expansion, while the persistent ratio denotes the fraction of update direction assigned to the persistent component. To this end, we propose two complementary mechanisms. Adaptive Koopman Control (AKC) regulates update magnitude by shrinking updates when recent prompt dynamics indicate expansion. Hankel Update Router (HUR) refines update direction by preserving persistent components across recent updates while suppressing oscillatory and residual components. These two mechanisms produce a stabilized prompt update. Since TPD only processes the base prompt update, it can be plugged into both visual and text TPT methods without modifying the backbone, prompt architecture, or adaptation loss. The contributions are summarized as follows:

- To the best of our knowledge, Test-Time Prompt-Agnostic Decomposition (TPD) is the first prompt-agnostic framework that analyzes and stabilizes prompt-update trajectories across ViT-based visual TPT and CLIP-based text TPT methods.
- We introduce Adaptive Koopman Control (AKC), which uses the spectral radius to scale down updates under expansive local dynamics and further attenuates them under consecutive unstable steps, preventing excessive prompt-state movement.

- We propose Hankel Update Router (HUR), which uses the three-component decomposition to reshape update directions by preserving persistent components and suppressing oscillatory and residual components, thereby mitigating directional conflict and cumulative drift.
- Extensive experiments on 15 datasets across shifted-domain and cross-dataset benchmarks demonstrate the generality of TPD. Across 10 TPT methods, TPD achieves the accuracy gains of 3–8% for visual prompts and over 2% for text prompts.

2 Related Work

Prompt-based test-time adaptation. Most existing TPT methods are developed for CLIP-based text prompt tuning and can be broadly divided into episodic and online modes. In episodic mode, prompts are adapted independently for each test sample or episode and are typically reset afterward, including initialization and calibration [Abdul Samadh et al., 2023, Yoon et al., 2024], sample filtering or augmentation selection [Shu et al., 2022, Feng et al., 2023, Farina et al., 2024], and black-box prompt update rules [Meng et al., 2025]. In online mode, prompts are updated continuously along test steps, and existing methods mainly regulate prompt drift or leverage prompt history [Ma et al., 2023, Zhang et al., 2024, Xiao et al., 2025]. Furthermore, CTPT [Wang et al., 2025a] focuses on continual prompt tuning across multiple target domains, aiming at prompt retention and generalization over newly learned streaming data. Some prompt-based methods are also built on ViT-based visual prompts [Jia et al., 2022]; for example, DePT [Gao et al., 2022] improves ViT prompt adaptation under domain adaptation with memory-bank and pseudo-label.

Despite these differences, existing TPT methods still intervene mainly at individual test steps. Moreover, CTPT [Wang et al., 2025a] targets continual multi-domain prompt retention, and DePT [Gao et al., 2022] is developed for domain adaptation. In contrast, our method analyzes and stabilizes cross-step prompt-update trajectories within each single target-domain adaptation process.

Prompt-based tuning and optimization. Prompt-based optimization has been developed across downstream tasks. Specifically, some methods improve prompt construction and backbone interaction [Xiao et al., 2026a]. VPT introduces learnable visual prompts for adapting a frozen ViT [Jia et al., 2022]; E2VPT extends visual prompt tuning with more effective prompt insertion across layers [Han et al., 2023]; and CVPT further strengthens prompt interaction across layers [Huang et al., 2025]. In vision-language models, ArGue improves prompt construction through attribute-guided prompts [Tian et al., 2024], while TCP uses class-aware textual prompts to better align prompts with class semantics [Yao et al., 2024]. Feature prompting [Cong et al., 2026b,a] and hierarchical image–text learning [Fan et al., 2026] have also been explored in medical imaging applications. Furthermore, some methods improve optimization and stabilization during training [Xiao et al., 2026b,c, Lan and Tian, 2024, Lan et al., 2026]. Fair-VPT introduces fairness-aware objectives to regularize prompt learning [Park and Byun, 2024], BVPT analyzes bursty update behavior during prompt optimization [Wang et al., 2025b]. PAE [Wang et al., 2026] is related to TPD in its use of dynamical systems, but it focuses on training-time visual prompt optimization by modeling cross-layer prompt evolution. In contrast, TPD analyzes and stabilizes the prompt-update trajectory of test samples.

These methods improve prompt learning through more effective training-time optimization, but prompt optimization under test-time inference remains largely unexplored. In contrast, our work focuses on prompt adaptation during test-time inference, with an emphasis on trajectory-level diagnosis and control of prompt updates across test steps.

3 Method

3.1 Test-Time Prompt-Agnostic Decomposition

Given unlabeled test inputs, let $f(\mathcal{B}_t; \theta, \mathcal{P}_t)$ denote a pretrained foundation model at test step t with frozen backbone θ, *e.g.*, ViT and CLIP. For ViT-based visual TPT, $\mathcal{P}_t$ consists of learnable prompt tokens injected into the encoder blocks, and $\mathcal{B}_t$ is the current mini-batch across test steps. For CLIP-based text TPT, $\mathcal{P}_t$ consists of learnable text prompt tokens used to construct class textual features, while $\mathcal{B}_t$ depends on the test-time adaptation (TTA) mode: in online mode, $\mathcal{B}_t$ is the current input that updates the text prompts cumulatively; in episodic mode, $\mathcal{B}_t$ is the current episode, typically a set of augmented views from a single test sample.

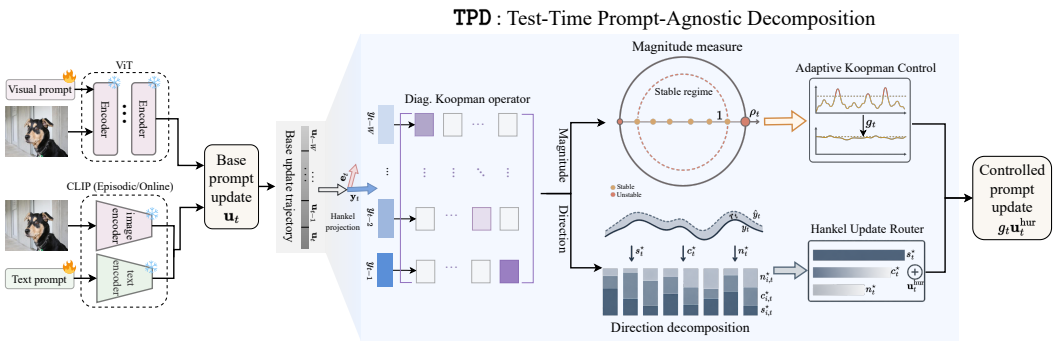


Figure 2: Overview of TPD. The base prompt update $\mathbf{u}_t$ from visual or text TPT is delay-embedded with a Hankel window and projected into a local Hankel subspace. AKC regulates update magnitude and HUR refines update direction. The controlled magnitude and refined direction are then combined into a stabilized prompt update $g_t\mathbf{u}_t^{\text{hur}}$, which is applied to the prompt state at step t.

For each ViT- or CLIP-based TPT method, we flatten its prompt tokens into a single vector:

$$\mathbf{p}_t = \text{vec}(\mathcal{P}_t) \in \mathbb{R}^d, \tag{1}$$

where vec concatenates the prompt tokens, and d is the prompt-vector dimension. Given $\mathcal{B}_t$ and a TTA loss $\mathcal{L}_{\text{tta}}$, which is typically employed as entropy minimization [Wang et al., 2020], the base prompt-update $\mathbf{u}_t$ at step t is

$$\mathbf{u}_t = -\eta_t \phi_t(\nabla_{\mathbf{p}} \mathcal{L}_{\text{tta}}(\mathcal{B}_t; \mathbf{p}_t)). \tag{2}$$

where η_t is the learning rate at step t, ϕ_t denotes the base optimizer, and $\nabla_{\mathbf{p}} \mathcal{L}_{\text{tta}}(\mathcal{B}_t; \mathbf{p}_t)$ denotes the gradient of the TTA loss with respect to the prompt state $\mathbf{p}_t$.

By taking the base prompt-update $\mathbf{u}_t$ as input and stabilizing it directly, TPD is agnostic to visual and textual prompt types. As shown in Figure 2, TPD analyzes the recent prompt-update trajectory in a short-window subspace and decomposes it: the spectral radius captures magnitude expansion, while an additive decomposition separates persistent, oscillatory, and residual directional components.

Hankel subspace from recent updates. TPD builds a short-window representation of the recent prompt-update trajectory using the delay-coordinate construction in Hankel-DMD [Arbabi and Mezic, 2017]. Specifically, it uses the latest W base prompt updates and converts them into M overlapping Hankel embeddings, where each embedding stacks K consecutive updates and $W = K + M - 1$. For any index $\tau \geq K$, the Hankel-lifted vector $\mathbf{z}_\tau$ stacks the latest K update vectors: $\mathbf{z}_\tau = \left[\mathbf{u}_{\tau-K+1}^\top, \mathbf{u}_{\tau-K+2}^\top, \ldots, \mathbf{u}_\tau^\top\right]^\top \in \mathbb{R}^{Kd}$. Using the M embeddings before the current update, we define the Hankel history matrix at test step t:

$$\mathcal{H}_t = \left[\mathbf{z}_{t-M}, \mathbf{z}_{t-M+1}, \ldots, \mathbf{z}_{t-1}\right] \in \mathbb{R}^{Kd \times M}. \tag{3}$$

Let $Q_t \in \mathbb{R}^{Kd \times r}$ contain the top-r left singular vectors of $\mathcal{H}_t$[2], where $r \leq \min(Kd, M)$ is selected adaptively from the current singular spectrum, and project the current lifted update $\mathbf{z}_t$ onto this short-window Hankel subspace:

$$\mathbf{y}_t = Q_t^\top \mathbf{z}_t \in \mathbb{R}^r, \qquad \mathbf{z}_t = Q_t \mathbf{y}_t + \widetilde{\mathbf{e}}_t, \qquad \widetilde{\mathbf{e}}_t = (I_{Kd} - Q_t Q_t^\top)\mathbf{z}_t. \tag{4}$$

The prompt-update decomposition is obtained by extracting the most recent block:

$$\mathbf{u}_t = E_K Q_t \mathbf{y}_t + \mathbf{e}_t, \qquad \mathbf{e}_t = E_K \widetilde{\mathbf{e}}_t, \qquad E_K = [\mathbf{0}_{d \times (K-1)d} \;\; I_d], \tag{5}$$

where $\widetilde{\mathbf{e}}_t$ is the residual, and $\mathbf{e}_t$ is its most recent d-dimensional prompt-update block.

Short-window Koopman operator. To capture recent update dynamics in the projected space, we map the historical lifted updates in $\mathcal{H}_t$ with the projection matrix Q_t and construct

$$\begin{aligned} Y_t^- &= \left[Q_t^\top z_{t-M}, \ldots, Q_t^\top z_{t-2}\right] = \left[\mathbf{y}_{t-M}, \ldots, \mathbf{y}_{t-2}\right], \\ Y_t^+ &= \left[Q_t^\top z_{t-M+1}, \ldots, Q_t^\top z_{t-1}\right] = \left[\mathbf{y}_{t-M+1}, \ldots, \mathbf{y}_{t-1}\right]. \end{aligned} \tag{6}$$

[2]By the Eckart–Young theorem [Eckart and Young, 1936], the top singular directions provide the optimal rank-r subspace.

Each column pair thus represents one recent update transition before test step t. We then fit a diagonal operator $A_t = \mathrm{diag}(\mathbf{a}_t)$ using scalar least-squares estimates:

$$a_{i,t} = \frac{\sum_{\tau=t-M}^{t-2} y_{i,\tau+1} y_{i,\tau}}{\sum_{\tau=t-M}^{t-2} y_{i,\tau}^2}, \qquad i = 1, \ldots, r, \qquad \mathbf{a}_t = [a_{1,t}, \ldots, a_{r,t}]^\top, \tag{7}$$

where $y_{i,\tau}$ is the i-th scalar entry of $\mathbf{y}_\tau$. The sign of $a_{i,t}$ distinguishes continuation from reversal, while its magnitude measures amplification in that coordinate.

Spectral radius: a magnitude signal. To quantify the magnitude expansion across steps, we compute the spectral radius $\rho_t = \rho(A_t) = \max |a_t|$. The larger ρ_t indicates stronger local magnitude expansion at step t. Since A_t is estimated only from projected lifted updates before step t, ρ_t uses no future information and can be monitored online.

Three-component decomposition: a direction signal. To analyze the direction of each prompt update, `TPD` leverages Dynamic Mode Decomposition (DMD) [Schmid, 2010] under the local Koopman model and decomposes each projected update into three additive components. Let $\hat{\mathbf{y}}_t = A_t \mathbf{y}_{t-1}$ be the prediction and $\mathbf{r}_t = \mathbf{y}_t - \hat{\mathbf{y}}_t$ be the prediction residual. From the Koopman operator A_t and its prediction $\hat{\mathbf{y}}_t$, we derive two element-wise gates

$$\mathbf{q}_t = \frac{\hat{\mathbf{y}}_t^{\,2}}{\hat{\mathbf{y}}_t^{\,2} + \mathbf{r}_t^{\,2}}, \qquad \boldsymbol{\pi}_t = \frac{1 + \tanh(\mathbf{a}_t)}{2}, \tag{8}$$

where $\mathbf{q}_t \in [0,1]^r$ measures how well the local Koopman model explains $\mathbf{y}_t$, and $\boldsymbol{\pi}_t \in [0,1]^r$ distinguishes same-sign continuation from sign flipping, recovering the masks $\mathbb{1}[\mathbf{a}_t > 0]$ and $\mathbb{1}[\mathbf{a}_t < 0]$. Using these gates, the projected update admits the decomposition

$$\mathbf{y}_t = \underbrace{\mathbf{q}_t \boldsymbol{\pi}_t \mathbf{y}_t}_{\mathbf{s}_t^\star} + \underbrace{\mathbf{q}_t (1 - \boldsymbol{\pi}_t) \mathbf{y}_t}_{\mathbf{c}_t^\star} + \underbrace{(1 - \mathbf{q}_t) \mathbf{y}_t}_{\mathbf{n}_t^\star}. \tag{9}$$

The persistent component $s_t^\star$ receives weight through both local predictability and same-sign continuation. The oscillatory component $c_t^\star$ captures locally explained reversal, while $n_t^\star$ contains content unexplained by the fitted dynamics. These are operational descriptions of the recent update trajectory. Oscillatory and residual components may still contain useful corrections, which motivates retaining them with nonzero routing weights (see Appendix A for justification).

3.2 Adaptive Koopman Control

Given the magnitude signal ρ_t from `TPD`, `AKC` modulates the prompt-update magnitude through a multiplicative gain factor g_t applied at each step:

$$g_t = \begin{cases} 1, & \rho_t \le \bar{\rho}, \\ \max\left(g_{\min}, \dfrac{\bar{\rho}}{\rho_t}\right), & \rho_t > \bar{\rho}, \end{cases} \tag{10}$$

where $\bar{\rho}$ is a stability threshold and $g_{\min} \in (0,1)$ is a lower bound on the gain. `AKC` preserves the update when $\rho_t \le \bar{\rho}$ and scales it down by $g_t < 1$ when $\rho_t > \bar{\rho}$. Across consecutive unstable steps, this scaling keeps the applied updates small while the local dynamics remain unstable (see Lemma 4 for the comparison between diagonal and full modes).

Lemma 1 (`AKC` non-amplification)**.** *Let A_t be the local Koopman operator in Eq.* (7) *and g_t the gain factor in Eq.* (10)*. Then $\rho(g_t A_t) = g_t\, \rho(A_t) \le \rho(A_t)$, with equality if and only if $\rho_t \le \bar{\rho}$. Moreover,*

$$\rho(g_t A_t) \;\le\; \max\big(\bar{\rho},\; g_{\min} \rho_t\big), \tag{11}$$

and in particular $\rho(g_t A_t) \le \bar{\rho}$ whenever $\rho_t \le \bar{\rho}/g_{\min}$.

Proof sketch. *g_t scales the eigenvalues of A_t, and the remaining bounds follow by splitting the gain definition in Eq.* (10)*. The gain floor prevents update shutdown, but it also means that the controlled spectral radius is not always below $\bar{\rho}$; the applicable bound is Eq.* (11)*.*

3.3 Hankel Update Router

Given the TPD three-component decomposition, HUR refines the update direction in the Hankel-lifted space and then extracts the most recent prompt-update block. Let $\mathbf{s}_t^\star, \mathbf{c}_t^\star, \mathbf{n}_t^\star \in \mathbb{R}^r$ collect the coordinate-wise persistent, oscillatory, and unexplained components. Let $\widetilde{\mathbf{e}}_t = (I_{Kd} - Q_t Q_t^\top)\mathbf{z}_t$ be the residual in the Hankel-lifted space and $\mathbf{e}_t = E_K \widetilde{\mathbf{e}}_t$ its extracted prompt-update block. We define the lifted components:

$$\widetilde{\mathbf{S}}_t = Q_t \mathbf{s}_t^\star, \qquad \widetilde{\mathbf{C}}_t = Q_t \mathbf{c}_t^\star, \qquad \widetilde{\mathbf{N}}_t = Q_t \mathbf{n}_t^\star + \widetilde{\mathbf{e}}_t. \tag{12}$$

We treat both the projected unexplained component $\mathbf{n}_t^\star$ and the residual $\widetilde{\mathbf{e}}_t$ as residual content. Their prompt-update-space counterparts are

$$\mathbf{S}_t = E_K \widetilde{\mathbf{S}}_t, \qquad \mathbf{C}_t = E_K \widetilde{\mathbf{C}}_t, \qquad \mathbf{N}_t = E_K \widetilde{\mathbf{N}}_t = E_K Q_t \mathbf{n}_t^\star + \mathbf{e}_t. \tag{13}$$

Thus $\mathbf{u}_t = \mathbf{S}_t + \mathbf{C}_t + \mathbf{N}_t$, where $\mathbf{S}_t, \mathbf{C}_t, \mathbf{N}_t \in \mathbb{R}^d$ are additive components induced by the projected Hankel decomposition. HUR routes the prompt-space components by preserving the persistent component and attenuating the oscillatory and residual components:

$$\mathbf{u}_t^{\mathrm{hur}} = \mathbf{S}_t + \beta_c \mathbf{C}_t + \beta_n \mathbf{N}_t, \qquad 0 \le \beta_n \le \beta_c \le 1, \tag{14}$$

where β_c controls the oscillatory component and β_n controls the residual component. Let $\widetilde{\mathbf{z}}_t^{\mathrm{hur}}$ denote the corresponding lifted routed update, obtained by applying the same routing in the Hankel-lifted space before extracting $\mathbf{u}_t^{\mathrm{hur}} = E_K \widetilde{\mathbf{z}}_t^{\mathrm{hur}}$. In the lifted space, HUR reweights each projected coordinate by a scalar in $[\beta_n, 1]$ and scales the residual by β_n, thereby attenuating selected components without reversing projected coordinates or increasing their magnitude.

3.4 Controlled Prompt Update

Combining the magnitude control of AKC with the direction refinement of HUR, TPD converts the base prompt-update into a controlled prompt-update:

$$\mathbf{u}_t \;\longrightarrow\; \Delta\mathbf{p}_t = g_t \mathbf{u}_t^{\mathrm{hur}}, \tag{15}$$

where $\Delta\mathbf{p}_t$ denotes the controlled prompt update applied to the prompts at the current test step t. Finally, Lemma 1 and the routing form in Eq. (14) give the following controlled-update properties.

Theorem 1 (Controlled prompt-update properties)**.** *Whenever the local Hankel window is available and* TPD *is applied, the controlled update* $\Delta\mathbf{p}_t = g_t \mathbf{u}_t^{\mathrm{hur}}$ *in Eq.* (15) *satisfies:*

(i) ***Hankel-lifted space non-amplification:*** $\|\widetilde{\mathbf{z}}_t^{\mathrm{hur}}\| \le \|\mathbf{z}_t\|$*, therefore* $\|\Delta\mathbf{p}_t\| \le g_t \|\mathbf{z}_t\| \le \|\mathbf{z}_t\|$.

(ii) ***Mode-wise routing without reversal:*** *each projected Hankel coordinate is reweighted as* $w_{i,t} y_{i,t}$ *for some* $w_{i,t} \in [\beta_n, 1]$*, so no projected coordinate is reversed or amplified.*

(iii) ***Persistent preservation in*** HUR ***routing:*** *within the defined Hankel decomposition, the persistent component* $\widetilde{\mathbf{S}}_t$ *and its prompt-space extraction* $\mathbf{S}_t$ *are transmitted with coefficient* 1*, while the oscillatory and residual components are attenuated by factors* β_c *and* β_n*.*

Proof sketch. *The proof follows from the weights* $w_{i,t} \in [\beta_n, 1]$ *in the Hankel subspace, the decomposition* $\mathbf{z}_t = Q_t \mathbf{y}_t + \widetilde{\mathbf{e}}_t$*, and the bound* $\|E_K\|_2 = 1$ *(see Appendix C for the full proof).*

Since both modules act directly on the prompt update $\mathbf{u}_t$, TPD can be attached to different TPT methods while preserving their original update rule. Together, AKC and HUR regulate the prompt-update trajectory by combining magnitude control with component-wise direction routing.

4 Experiments

4.1 Experimental Setup

Datasets. Following previous TPT protocols [Shu et al., 2022], we evaluate TPD on image classification under two distribution-shift settings, using ImageNet-pretrained backbones as source models and adapting them at test time on target datasets. The domain generalization benchmark includes

Table 1: Comparisons on the domain generalization setting with visual and text TPT methods. OOD Mean averages the 4 variants excluding ImageNet, while Mean averages ImageNet and the 4 variants. Values are the baseline accuracy with TPD gains.

Method	ImageNet	ImageNet-V2	ImageNet-S	ImageNet-A	ImageNet-R	*OOD Mean*	*Mean*
ViT-based visual prompt tuning							
ViT-B/16	81.44	70.85	35.44	30.92	47.48	46.17	53.23
VPT +TPD	78.16 **+3.15**	64.55 **+5.32**	32.77 **+3.25**	19.17 **+19.81**	42.14 **+4.33**	39.66 **+8.18**	47.36 **+7.17**
E2VPT +TPD	78.35 **+2.69**	64.09 **+4.93**	32.89 **+2.70**	18.17 **+20.79**	42.07 **+3.94**	39.31 **+8.09**	47.11 **+7.01**
VFPT +TPD	78.71 **+2.62**	65.34 **+4.75**	33.16 **+2.81**	22.05 **+18.68**	42.81 **+3.55**	40.84 **+7.45**	48.41 **+6.48**
BPT +TPD	81.30 **+2.87**	70.48 **+2.45**	28.62 **+9.93**	38.97 **+4.23**	43.68 **+5.39**	45.44 **+5.51**	52.61 **+4.97**
CLIP-based text prompt tuning							
CLIP	66.73	60.86	46.09	47.87	73.98	57.20	59.11
TPT +TPD	68.98 **+2.21**	63.45 **+2.53**	47.94 **+1.68**	54.77 **+1.96**	77.06 **+2.92**	60.81 **+2.27**	62.44 **+2.26**
C-TPT +TPD	69.28 **+2.35**	63.36 **+1.83**	48.55 **+2.02**	52.93 **+1.67**	78.97 **+3.85**	60.95 **+2.35**	62.62 **+2.34**
R-TPT +TPD	60.92 **+5.08**	64.91 **+2.61**	45.03 **+2.36**	48.92 **+2.48**	77.63 **+3.01**	59.12 **+2.62**	59.48 **+3.11**
O-TPT +TPD	67.33 **+4.52**	61.65 **+1.96**	47.12 **+1.81**	49.87 **+1.74**	72.55 **+4.59**	57.80 **+2.53**	59.70 **+2.92**
CAT +TPD	69.32 **+2.50**	60.91 **+2.34**	52.08 **+2.23**	58.69 **+2.51**	76.76 **+3.32**	62.11 **+2.60**	63.55 **+2.58**
DyPt +TPD	69.61 **+2.55**	64.63 **+2.47**	48.22 **+2.18**	56.16 **+2.39**	78.15 **+3.73**	61.79 **+2.69**	63.35 **+2.66**

Table 2: Comparisons on the cross-dataset setting with visual and text TPT methods. All methods are evaluated on 10 downstream datasets without training. Values are baseline accuracy with TPD gains.

Method	Caltech101	OxfordPets	StanfordCars	Flowers102	Food101	Aircraft	SUN397	DTD	EuroSAT	UCF101	*Mean*
ViT-based visual prompt tuning											
ViT-B/16	86.71	92.20	57.84	66.23	82.78	22.06	56.07	48.19	49.91	65.08	62.71
VPT +TPD	92.65 **+2.72**	86.89 **+2.49**	60.68 **+2.13**	64.86 **+3.23**	85.14 **+1.31**	21.21 **+3.25**	59.92 **+3.87**	38.19 **+4.06**	20.91 **+11.26**	59.28 **+2.73**	58.97 **+3.71**
E2VPT +TPD	92.29 **+2.01**	86.89 **+2.75**	60.03 **+2.08**	63.85 **+3.96**	84.55 **+1.26**	20.79 **+3.51**	59.07 **+3.48**	37.93 **+5.33**	17.29 **+11.67**	58.75 **+2.81**	58.14 **+3.89**
VFPT +TPD	92.76 **+2.36**	87.33 **+2.71**	62.01 **+2.41**	67.77 **+1.98**	85.18 **+1.43**	21.69 **+3.33**	59.31 **+2.64**	40.53 **+3.43**	23.54 **+12.66**	60.26 **+1.43**	60.04 **+3.44**
BPT +TPD	79.51 **+13.89**	68.26 **+14.45**	49.88 **+11.15**	64.82 **+2.33**	67.45 **+15.21**	21.47 **+4.15**	48.50 **+7.84**	41.46 **+1.96**	57.42 **+1.71**	50.52 **+9.59**	54.93 **+8.23**
CLIP-based text prompt tuning											
CLIP	93.35	88.25	65.48	67.44	83.65	23.67	62.59	44.27	42.01	65.13	63.58
TPT +TPD	94.16 **+2.16**	87.79 **+2.52**	66.87 **+1.68**	68.98 **+2.08**	84.67 **+3.65**	24.78 **+2.41**	65.50 **+3.01**	47.75 **+4.23**	42.44 **+3.59**	68.04 **+3.96**	65.10 **+2.93**
C-TPT +TPD	93.62 **+1.93**	88.23 **+2.27**	65.82 **+1.31**	69.84 **+2.51**	83.73 **+4.62**	24.02 **+2.39**	64.83 **+3.28**	46.04 **+2.49**	43.21 **+2.87**	65.74 **+1.48**	64.51 **+2.52**
R-TPT +TPD	93.67 **+2.09**	87.24 **+2.43**	67.03 **+1.34**	68.71 **+2.13**	83.94 **+4.02**	23.92 **+2.27**	64.37 **+3.61**	46.43 **+3.26**	34.72 **+3.08**	67.23 **+2.66**	63.73 **+2.69**
O-TPT +TPD	93.95 **+2.22**	87.95 **+2.38**	64.53 **+1.24**	70.07 **+2.24**	84.13 **+4.18**	23.64 **+2.17**	64.23 **+3.49**	45.68 **+3.30**	42.84 **+3.15**	64.16 **+2.58**	64.12 **+2.70**
CAT +TPD	94.72 **+2.36**	90.34 **+2.65**	69.41 **+1.52**	72.73 **+2.43**	86.24 **+4.41**	24.13 **+2.52**	67.02 **+3.88**	47.24 **+3.59**	50.53 **+3.46**	69.52 **+2.96**	67.19 **+2.98**
DyPt +TPD	94.32 **+2.28**	88.28 **+2.61**	67.65 **+1.43**	69.95 **+2.18**	85.42 **+4.08**	24.33 **+2.36**	66.32 **+3.72**	47.96 **+3.64**	42.28 **+3.22**	68.72 **+3.08**	65.52 **+2.86**

ImageNet [Deng et al., 2009] and four shifted variants: ImageNet-V2 [Recht et al., 2019], ImageNet-Sketch [Wang et al., 2019], ImageNet-A [Hendrycks et al., 2021b], and ImageNet-R [Hendrycks et al., 2021a]. The cross-dataset benchmark contains 10 downstream datasets covering fine-grained recognition, textures, scenes, satellite images, and actions (see Appendix D.1 for the full details).

Baselines. We attach TPD to TPT methods on two backbone families. For ViT-based visual prompt tuning, we evaluate four methods: VPT [Jia et al., 2022], E2VPT [Han et al., 2023], VFPT [Zeng et al., 2024], BPT [Wang et al., 2025b]. For CLIP-based text prompt tuning, we evaluate both episodic and online methods: TPT [Shu et al., 2022], C-TPT [Yoon et al., 2024], R-TPT [Sheng et al., 2025], O-TPT [Sharifdeen et al., 2025], CAT [Zhang et al., 2025], DyPt [Xiao et al., 2025]. We report both the original result and its gain with TPD, denoted as "Baseline +TPD".

Implementation Details. We use $K = 3$ updates per Hankel embedding and $M = 8$ embeddings, giving $W = 10$ past updates and seven transition pairs. These short windows balance temporal context with local estimation. The rank r is selected adaptively from the current singular spectrum. We set $\bar{\rho} = 1$, $g_{\min} = 0.1$, and $(\beta_c, \beta_n) = (0.25, 0.1)$ across all datasets, methods, and backbone families. The spectral threshold corresponds to the unit-circle boundary of the local modes; the gain floor avoids update shutdown, and the routing weights retain nonzero contributions from oscillatory and residual content. All experiments use an NVIDIA H100 GPU (see Appendix D.3 for per-dataset hyperparameter experimental results).

4.2 Results

For ViT-based visual TPT methods, TPD consistently improves all baselines under both domain generalization and cross-dataset transfer, as shown in Table 1 and Table 2. The shifted-domain OOD mean gains range from 5.51-8.18%, while the cross-dataset mean gains range from 3.44-8.23%. The improvements from adding TPD are especially large under hard shifts; for example, VPT gains 19.81% on ImageNet-A, and BPT gains 8.23% on average across cross-dataset transfer. This suggests that training-time visual prompts remain vulnerable when reused for naive test-time entropy updates, where update magnitudes may expand and directions may oscillate or drift. These larger gains therefore reflect the lack of dedicated TTA stabilizers for ViT prompt tuning: TPD supplies the missing trajectory control and turns otherwise unstable test-time tuning into effective adaptation.

For CLIP-based text TPT methods, TPD also improves all episodic and online baselines in Table 1 and Table 2. On domain generalization, the OOD mean gains are consistently between 2.27% and 2.69%; on cross-dataset transfer, the mean gains range from 2.52% to 2.98%. For example, DyPt improves from 61.79% to 64.48% in the domain generalization OOD mean. Although these gains are more moderate than those for visual prompts, they remain highly consistent across different adaptation designs. By acting directly on the base update vectors, TPD stabilizes text-prompt trajectories without replacing the original text-prompt objective. This consistency further shows that TPD serves as a prompt-agnostic module complementary to text TPT methods.

We further compare TPD with simple stabilization rules in Table 3, including update clipping [Pascanu et al., 2013], EMA updates [Tarvainen and Valpola, 2017], norm control [Niu et al., 2023], and low-pass filtering [Zhang et al., 2019]. These heuristics improve the TPT baselines but remain below TPD: on VPT, low-pass filtering improves DG from 39.66 to 44.28, while TPD reaches 47.84; on TPT, it improves CSD from 65.10 to 66.23, whereas TPD reaches 68.03. TPD also incurs modest overhead, with $1.13\times$ on VPT and $1.08\times$ on TPT. These results suggest that simple scale control or short-term smoothing is insufficient, while TPD further uses component-wise routing to stabilize prompt-update dynamics across both visual and text prompts under TTA (see Appendix D.2.1 for the per-dataset simple-stabilization results).

Table 3: Comparison with simple stabilization methods on visual and text TPT methods. DG and CSD denote the accuracy means over four ImageNet variants and 10 cross-datasets. $\times$: per-step wall-clock overhead relative to the baseline.

	VPT			TPT		
Variant	*DG*	*CSD*	$\times$	*DG*	*CSD*	$\times$
Baseline	39.66	58.97	1.00	60.81	65.10	1.00
Simple stabilization rules						
+ Update clipping	41.92	59.54	1.11	61.36	65.47	1.06
+ EMA update	42.74	59.86	0.99	61.65	65.82	1.01
+ Norm control	43.65	61.03	1.07	61.72	65.74	1.03
+ Low-pass filtering	44.28	60.91	1.00	62.23	66.23	1.00
Prompt-update decomposition and stabilization						
+TPD	47.84	62.68	1.13	63.08	68.03	1.08

4.3 Ablation Study

We conduct ablations on ViT-based VPT and CLIP-based TPT to examine three design choices in TPD: the Koopman operator for AKC, the projection matrix for HUR, and the routing weights in HUR. The first block compares a full Koopman mode with the proposed diagonal mode in Table 4. The diagonal mode performs better, improving VPT-DG from 43.58 to 45.92 and TPT-CSD from 66.38 to 67.21. This shows the short Hankel window makes cross-step coupling difficult to estimate reliably, while the diagonal mode provides stable scalar coefficients for spectral control.

The second block compares plain projection with the Hankel embedding used by HUR. Hankel projection consistently outperforms plain projection, improving VPT-DG from 42.31 to 45.98 and TPT-CSD from 66.01 to 67.54, indicating that delay embedding captures cross-step update dynamics missed by a static projection. Furthermore, the third block ablates HUR routing: w/o Pres., w/o Osc., and w/o Res. use $(\beta_c, \beta_c, \beta_n)$, $(1, 1, \beta_n)$, and $(1, \beta_c, 1)$, respectively,

Table 4: Ablation study of TPD on visual and text TPT methods. DG and CSD denote the accuracy means over four ImageNet variants and 10 cross-datasets.

	VPT		TPT	
Variant	*DG*	*CSD*	*DG*	*CSD*
Baseline	39.66	58.97	60.81	65.10
Ablation study of Koopman operator modes				
Full	43.58	60.86	61.80	66.38
Diagonal (+AKC)	45.92	61.74	62.56	67.21
Ablation study of projection matrix modes				
Plain	42.31	60.22	61.43	66.01
Hankel (+HUR)	45.98	62.18	62.84	67.54
Ablation study of HUR routing				
w/o Pres.	47.06	62.29	62.95	67.73
w/o Osc.	47.28	62.37	62.84	67.87
w/o Res.	46.84	62.21	62.80	67.54
Full				
+TPD	47.84	62.68	63.08	68.03

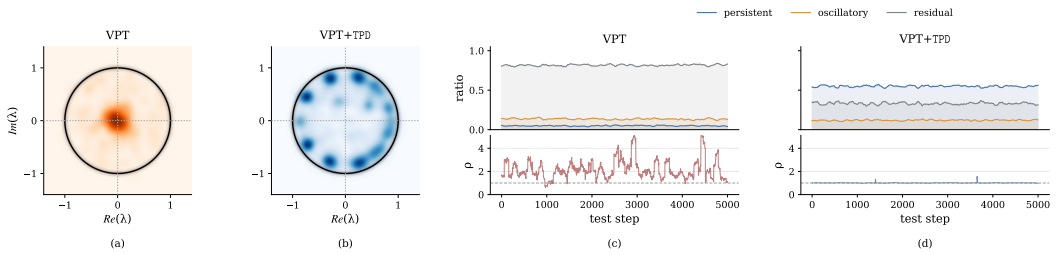


Figure 3: (a), (b) show Koopman spectra of visual prompt trajectories. The spectra are estimated post-hoc from generated update trajectories, and the black circle denotes the unit circle. (c), (d) show component ratios and spectral radius ρ over test steps.

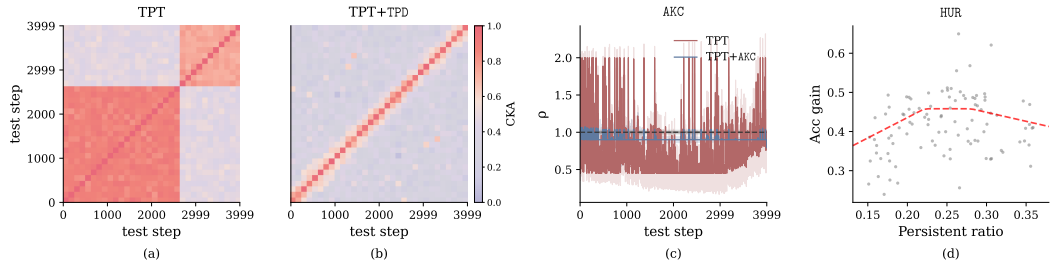


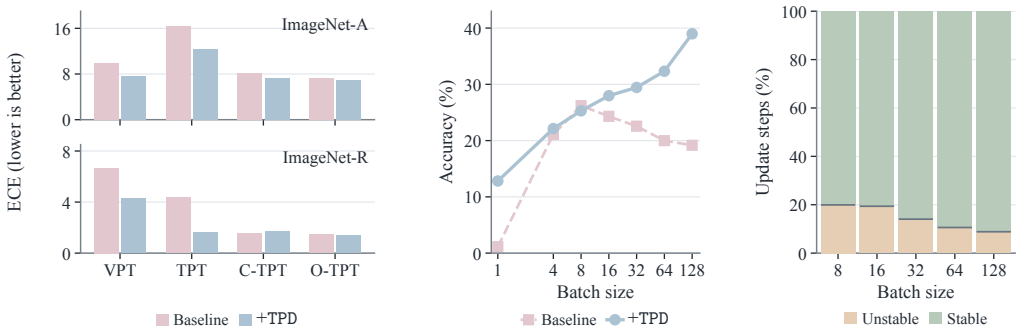
Figure 4: (a), (b) CKA similarity between text-prompt representations across test steps after smoothing. (c) Koopman spectral radius estimated from prompt updates, the estimate is post-hoc for TPT and used online by AKC. (d) Persistent ratio of HUR versus subsequent accuracy gain. Each gray point denotes one short temporal window. The red dashed curve is a piecewise linear fitted trend line.

instead of the full $(1, \beta_c, \beta_n)$. All three variants underperform TPD across VPT and TPT, supporting the routing strategy that preserves persistent directions, softly attenuates oscillatory directions, and strongly suppresses unexplained perturbations (see Appendix D.2.2 for the full ablation results).

4.4 Analysis

To quantify the stability of visual prompt-update trajectories, we visualize the Koopman spectra and decomposition dynamics in Figure 3. VPT concentrates most spectral mass near the origin, indicating that many update modes decay rapidly rather than carrying persistent adaptation. This does not contradict the large ρ values in Figure 3(c), since the spectrum aggregates eigenvalue density over the trajectory, while ρ tracks the largest local eigenvalue at each step. Thus, a trajectory can be dominated by decaying modes overall but still contain intermittent expansive steps. Consistently, VPT shows residual-dominated directions and frequent excursions of ρ beyond the stable reference, suggesting that its updates are both noisy in direction and unstable in magnitude. After applying TPD, the spectrum shifts toward the unit circle and becomes more evenly distributed, indicating that useful modes are preserved. Meanwhile, the persistent ratio increases, oscillatory and residual ratios decrease, and ρ stays closer to the stable regime. These results show that HUR improves the directional composition of updates, while AKC controls their magnitude expansion.

We further analyze text TPT methods after adding TPD. The CKA [Kornblith et al., 2019] matrices measure pairwise similarity between text-prompt representations across test steps. In Figure 4(a), TPT forms coarse high-similarity blocks, indicating that prompts collapse into a few similar states within local segments and change abruptly across segments. In contrast, TPT+TPD preserves a sharper near-diagonal structure in Figure 4(b), suggesting smoother stepwise evolution without excessive cross-step collapse. Figure 4(c) further shows that the Koopman spectral radius of TPT fluctuates strongly and frequently exceeds $\rho = 1$, whereas AKC keeps the radius closer to the stable reference. Finally, Figure 4(d) links the direction signal to performance: accuracy gain generally increases from low to moderate persistent ratios and then reaches a peak. This trend supports the use of persistent ratio as a routing signal for HUR, since low persistence corresponds to oscillatory or residual-dominated updates, while moderate persistence reflects useful adaptation structure (see Appendix D.4 for the visualizations on all datasets).



(a) Calibration (b) Batch sensitivity (c) TPD stability

Figure 5: (a) ECE on ImageNet-A and ImageNet-R, with separate vertical scales. (b) Accuracy across batch sizes. Pink and blue in (a,b) denote each baseline and its TPD variant, respectively. (c) Proportions of stable ($\rho_t \leq 1$, light green) and unstable ($\rho_t > 1$, light orange) TPD update steps, where ρ_t denotes the estimated spectral radius; each stacked bar sums to 100%.

4.5 Robustness and Reliability

We evaluate calibration across four prompt-tuning methods on ImageNet-A and ImageNet-R (Fig. 5(a)). TPD reduces expected calibration error (ECE) in seven of eight comparisons, with C-TPT on ImageNet-R as the only exception. For TPT, ECE decreases from 16.42% to 12.27% on ImageNet-A and from 4.36% to 1.62% on ImageNet-R. These improvements suggest that controlling prompt dynamics also regularizes predictive confidence, extending the benefits of stabilization to the alignment between confidence and empirical accuracy across multiple prompt-tuning formulations.

We examine VPT batch-size sensitivity on ImageNet-A with top-10% confidence filtering (Fig. 5(b)). TPD improves accuracy in six of seven settings, with a decrease at $B = 8$. At $B = 1$, accuracy increases from 1.14% to 12.82%, and at $B = 128$, from 19.17% to 38.98%. TPD improves over the baseline in most small-batch settings, although absolute accuracy remains low at very small batch sizes. Moreover, baseline accuracy declines beyond $B = 8$, while TPD accuracy continues to improve with increasing batch size. This divergent trend highlights the importance of update regulation in converting a larger sample budget into more effective adaptation.

We further assess the proportions of spectrally stable and unstable TPD updates (Fig. 5(c)). The stable fraction increases from 80.07% at $B = 8$ to 91.16% at $B = 128$, with a consistent upward trend across the five measured batch sizes. This pattern suggests that larger batches provide a stronger basis for estimating and regulating local dynamics, reducing the frequency of unstable update behavior. Viewed alongside the accuracy trends, it supports a link between the reliability of local dynamics and adaptation quality. The stable fraction complements accuracy by characterizing the update process itself, allowing predictive performance and dynamical reliability to be assessed separately (see Appendices D.5–D.8 for results on nonstationary streams, component ratios, seed variation, and adaptive magnitude controllers).

5 Conclusion

We propose TPD, which characterizes observed prompt-update trajectories from a dynamical-systems view and decomposes base prompt updates into magnitude and direction. Specifically, TPD projects recent updates into a Hankel subspace, AKC constrains unstable update magnitudes, and HUR preserves persistent components while attenuating oscillatory and residual components. As a prompt-agnostic framework, TPD operates directly on base prompt updates and serves as a plug-in stabilizer without modifying the backbone, prompt architecture, or adaptation loss. Across visual and text prompts, TPD consistently improves TPT baselines under domain generalization and cross-dataset settings.

Limitation. TPD relies on a meaningful recent update history and may be less effective when updates are sparse, noisy, or rapidly changing. Its finite-window local model does not guarantee global convergence or semantic correctness of retained directions, and soft routing may attenuate useful corrections. Small-batch adaptation also remains challenging.

References

Jameel Abdul Samadh, Mohammad Hanan Gani, Noor Hussein, Muhammad Uzair Khattak, Muhammad Muzammal Naseer, Fahad Shahbaz Khan, and Salman H Khan. Align your prompts: Test-time prompting with distribution alignment for zero-shot generalization. *Advances in Neural Information Processing Systems*, 36:80396–80413, 2023.

Hassan Arbabi and Igor Mezic. Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the koopman operator. *SIAM Journal on Applied Dynamical Systems*, 16(4): 2096–2126, 2017.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European Conference on Computer Vision*, pages 446–461. Springer, 2014.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3606–3613, 2014.

Cong Cong, Yang Song, Antonio Di Ieva, Angela Chou, Anthony J. Gill, Enrico Coiera, and Sidong Liu. Flexible and scalable federated learning with deep feature prompts for digital pathology. *npj Digital Medicine*, 2026a.

Cong Cong, Yang Song, Antonio Di Ieva, Qiangguo Jin, Lei Fan, Angela Chou, Anthony J Gill, and Sidong Liu. Slide-aware deep feature prompting for enhanced whole slide image classification. *Expert Systems with Applications*, page 133079, 2026b.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

Lei Fan, Arcot Sowmya, Erik Meijering, Zhen Yu, Zongyuan Ge, and Yang Song. Medical hierarchical image classification via dual-geometry image–text learning. *Medical Image Analysis*, page 104120, 2026.

Matteo Farina, Gianni Franchi, Giovanni Iacca, Massimiliano Mancini, and Elisa Ricci. Frustratingly easy test-time adaptation of vision-language models. *Advances in Neural Information Processing Systems*, 37:129062–129093, 2024.

Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop*, pages 178–178. IEEE, 2004.

Chun-Mei Feng, Kai Yu, Yong Liu, Salman Khan, and Wangmeng Zuo. Diverse data augmentation with diffusions for effective test-time prompt tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2704–2714, 2023.

Yunhe Gao, Xingjian Shi, Yi Zhu, Hao Wang, Zhiqiang Tang, Xiong Zhou, Mu Li, and Dimitris N Metaxas. Visual prompt tuning for test-time domain adaptation. *arXiv preprint arXiv:2210.04831*, 2022.

Cheng Han, Qifan Wang, Yiming Cui, Zhiwen Cao, Wenguan Wang, Siyuan Qi, and Dongfang Liu. E^ 2vpt: An effective and efficient approach for visual prompt tuning. *arXiv preprint arXiv:2307.13770*, 2023.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *IEEE International Conference on Computer Vision*, pages 8340–8349, 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 15262–15271, 2021b.

Lingyun Huang, Jianxu Mao, Junfei Yi, Ziming Tao, and Yaonan Wang. Cvpt: Cross visual prompt tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 848–858, 2025.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European conference on computer vision*, pages 709–727. Springer, 2022.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMlR, 2019.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshops*, pages 554–561, 2013.

Qizhen Lan and Qing Tian. Gradient-guided knowledge distillation for object detectors. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 423–432. IEEE, 2024.

Qizhen Lan, Xi Xiao, Xiangchen Guan, Mengchen Fan, Moule Lin, Jung Im Choi, and Lijing Zhu. Trust is not enough: Influence calibration for on-policy self-distillation in agentic rl. *arXiv preprint arXiv:2608.14945*, 2026.

Jian Liang, Ran He, and Tieniu Tan. A comprehensive survey on test-time adaptation under distribution shifts. *International Journal of Computer Vision*, 133(1):31–64, 2025.

Xiaosong Ma, Jie Zhang, Song Guo, and Wenchao Xu. Swapprompt: Test-time prompt adaptation for vision-language models. *Advances in Neural Information Processing Systems*, 36:65252–65264, 2023.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Fan'an Meng, Chaoran Cui, Hongjun Dai, and Shuai Gong. Black-box test-time prompt tuning for vision-language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pages 722–729. IEEE, 2008.

Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. *arXiv preprint arXiv:2302.12400*, 2023.

Sungho Park and Hyeran Byun. Fair-vpt: Fair visual prompt tuning for image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12268–12278, 2024.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3498–3505. IEEE, 2012.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. Pmlr, 2013.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019.

Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

Ashshak Sharifdeen, Muhammad Akhtar Munir, Sanoojan Baliah, Salman Khan, and Muhammad Haris Khan. O-tpt: Orthogonality constraints for calibrating test-time prompt tuning in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19942–19951, 2025.

Lijun Sheng, Jian Liang, Zilei Wang, and Ran He. R-tpt: Improving adversarial robustness of vision-language models through test-time prompt tuning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29958–29967, 2025.

Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. *Advances in Neural Information Processing Systems*, 35:14274–14289, 2022.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.

Xinyu Tian, Shu Zou, Zhaoyuan Yang, and Jing Zhang. Argue: Attribute-guided prompt tuning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28578–28587, 2024.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020.

Fan Wang, Zhongyi Han, Xingbo Liu, Yilong Yin, and Xin Gao. Ctpt: Continual test-time prompt tuning for vision-language models. *Pattern Recognition*, 161:111300, 2025a.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Junze Wang, Lei Fan, Dezheng Zhang, Weipeng Jing, Donglin Di, Yang Song, Sidong Liu, and Cong Cong. Visual prompt-agnostic evolution. *arXiv preprint arXiv:2601.20232*, 2026.

Yuzhu Wang, Manni Duan, and Shu Kong. Attention to the burstiness in visual prompt tuning! In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4253–4263, 2025b.

Matthew O Williams, Ioannis G Kevrekidis, and Clarence W Rowley. A data–driven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015.

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3485–3492. IEEE, 2010.

Xi Xiao, Xingjian Li, Yunbei Zhang, Cheng Han, Tianming Liu, Tianyang Wang, Runmin Jiang, Jihun Hamm, Xiao Wang, and Min Xu. Layer-specific prompt fusion discovery via differentiable search in vision foundation models. *arXiv preprint arXiv:2606.26379*, 2026a.

Xi Xiao, Chen Liu, Chih-Ting Liao, Yunbei Zhang, Qizhen Lan, Yuxiang Wei, Lin Zhao, Janet Wang, Jianyang Gu, Muchao Ye, et al. Staying vigilant: Mitigating visual laziness via counterfactual visual alignment in mllms. *arXiv preprint arXiv:2606.26387*, 2026b.

Xi Xiao, Chenrui Ma, Yunbei Zhang, Chen Liu, Zhuxuanzi Wang, Yanshu Li, Lin Zhao, Guosheng Hu, Tianyang Wang, and Hao Xu. Not all directions matter: Towards structured and task-aware low-rank model adaptation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2132–2154, 2026c.

Zehao Xiao, Shilin Yan, Jack Hong, Jiayin Cai, Xiaolong Jiang, Yao Hu, Jiayi Shen, Qi Wang, and Cees GM Snoek. Dynaprompt: Dynamic test-time prompt tuning. *arXiv preprint arXiv:2501.16404*, 2025.

Hantao Yao, Rui Zhang, and Changsheng Xu. Tcp: Textual-based class-aware prompt tuning for visual-language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23438–23448, 2024.

Hee Suk Yoon, Eunseop Yoon, Joshua Tian Jin Tee, Mark Hasegawa-Johnson, Yingzhen Li, and Chang D Yoo. C-tpt: Calibrated test-time prompt tuning for vision-language models via text feature dispersion. *arXiv preprint arXiv:2403.14119*, 2024.

Longhui Yuan, Binhui Xie, and Shuang Li. Robust test-time adaptation in dynamic scenarios. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15922–15932, 2023.

Runjia Zeng, Cheng Han, Qifan Wang, Chunshu Wu, Tong Geng, Lifu Huang, Ying N Wu, and Dongfang Liu. Visual fourier prompt tuning. *Advances in Neural Information Processing Systems*, 37:5552–5585, 2024.

Jingyi Zhang, Jiaxing Huang, Xiaoqin Zhang, Ling Shao, and Shijian Lu. Historical test-time prompt tuning for vision foundation models. *Advances in Neural Information Processing Systems*, 37: 12872–12896, 2024.

Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. *Advances in neural information processing systems*, 32, 2019.

Youjia Zhang, Huiling Liu, Youngeun Kim, and Sungeun Hong. Cat-tpt: Class-agnostic text-based test-time prompt tuning for vision-language models. *International Journal of Computer Vision*, 133(10):6930–6952, 2025.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.

A Spectral Justification of the Three-Component Decomposition

This section justifies the three-component decomposition in Eq. (9) from the Hankel-DMD spectral view. The key point is that AKC and HUR use different aspects of the same local Koopman spectrum. The spectral radius $\rho(A_t)$ depends on eigenvalue magnitudes and diagnoses local magnitude expansion for AKC. In contrast, the persistent and oscillatory split depends on spectral phase, or on the sign of each scalar coefficient under the diagonal local model, and is used by HUR to route update directions.

Lemma 2 (Hankel-DMD spectral reconstruction)**.** *Consider the projected Hankel trajectory in the local subspace at test step t, and suppose that within the short window it is represented by a finite-dimensional local Koopman model*

$$\mathbf{y}_{\tau+1} \approx A_t \mathbf{y}_\tau. \tag{16}$$

If A_t is diagonalizable with eigenpairs $\{(\lambda_j, \boldsymbol{\phi}_j)\}_{j=1}^r$, then the local DMD reconstruction can be written as

$$\mathbf{y}_\tau \approx \sum_{j=1}^{r} b_j \boldsymbol{\phi}_j \lambda_j^\tau, \qquad \lambda_j = |\lambda_j| e^{\mathrm{i}\theta_j}, \qquad \omega_j = \log(\lambda_j) = \alpha_j + \mathrm{i}\beta_j. \tag{17}$$

Here $|\lambda_j|$ (or α_j) controls growth or decay, while θ_j (or β_j) controls phase rotation and hence oscillatory behavior.

Proof. The DMD reconstruction follows from expanding the initial projected state in the eigenbasis of A_t: if $\mathbf{y}_0 = \sum_j b_j \boldsymbol{\phi}_j$, then $A_t^\tau \mathbf{y}_0 = \sum_j b_j \boldsymbol{\phi}_j \lambda_j^\tau$. Writing $\lambda_j = |\lambda_j| e^{\mathrm{i}\theta_j}$ gives $\lambda_j^\tau = |\lambda_j|^\tau e^{\mathrm{i}\theta_j \tau}$, which separates magnitude evolution from phase evolution. Equivalently, using the continuous-time parameter $\omega_j = \log(\lambda_j) = \alpha_j + \mathrm{i}\beta_j$, each mode evolves as $e^{\omega_j \tau} = e^{\alpha_j \tau} e^{\mathrm{i}\beta_j \tau}$. Thus eigenvalue magnitude controls whether the mode grows or decays, while eigenvalue phase controls whether the mode rotates or oscillates. □

Lemma 3 (Spectral partition into persistent, oscillatory, and residual parts)**.** *For a real-valued local trajectory reconstructed by Eq.* (17)*, the retained Hankel-DMD modes admit the following partition.*

- *Modes with $\theta_j = 0$ have positive real eigenvalues and produce non-oscillatory exponential evolution. These modes correspond to persistent content.*
- *Modes with $\theta_j = \pi$ have negative real eigenvalues and produce alternating-sign dynamics. These modes correspond to oscillatory content.*
- *Non-real conjugate pairs with $\theta_j \notin \{0, \pi\}$ combine into exponentially modulated sine and cosine terms, and therefore correspond to oscillatory content.*
- *The part not explained by the retained Hankel-DMD reconstruction is residual content.*

Proof. For a positive real eigenvalue, $\lambda_j = |\lambda_j|$ and the corresponding scalar time factor is $|\lambda_j|^\tau = e^{\alpha_j \tau}$, which has no phase rotation and does not alternate sign. It is therefore non-oscillatory within the local model. For a negative real eigenvalue, $\lambda_j = -|\lambda_j| = |\lambda_j| e^{\mathrm{i}\pi}$, so $\lambda_j^\tau = |\lambda_j|^\tau (-1)^\tau$, yielding alternating-sign behavior across steps. For a non-real pair $\lambda_j, \overline{\lambda_j}$, the real-valued contribution of the pair is

$$b_j \boldsymbol{\phi}_j e^{(\alpha_j + \mathrm{i}\beta_j)\tau} + \overline{b_j \boldsymbol{\phi}_j}\, e^{(\alpha_j - \mathrm{i}\beta_j)\tau} = e^{\alpha_j \tau}\big(\mathbf{a}_j \cos(\beta_j \tau) + \mathbf{b}_j \sin(\beta_j \tau)\big), \tag{18}$$

for real vectors $\mathbf{a}_j, \mathbf{b}_j$. This is an exponentially modulated sinusoid and is therefore oscillatory. Finally, any discrepancy between the current projected update and the retained local DMD reconstruction is, by definition, unexplained by the retained modes and forms residual content. □

Connection to the diagonal implementation. The full spectral partition above explains the origin of the three components. The implementation in the main text uses the diagonal local model $A_t = \mathrm{diag}(\mathbf{a}_t)$, so each projected coordinate is a scalar local DMD mode. Accordingly, the products, divisions, powers, and nonlinear functions in Eqs. (7)–(9) are applied coordinate-wise. For coordinate

i, a positive coefficient $a_{i,t} > 0$ has phase 0 and represents same-sign, non-oscillatory continuation; a negative coefficient $a_{i,t} < 0$ has phase π and represents sign-flipping oscillation. The magnitude $|a_{i,t}|$ only describes local amplification or decay and is used by $\rho(A_t)$ for `AKC`; it does not determine whether the coordinate is persistent or oscillatory. This separates the magnitude branch from the direction branch.

The gates in Eq. (8) use these coordinate-wise quantities to assign the observed projected update in Eq. (9) to persistent, oscillatory, and residual parts. Specifically, $\boldsymbol{\pi}_t = (1 + \tanh(\mathbf{a}_t))/2$ softly separates same-sign continuation from sign-flipping dynamics, while

$$q_{i,t} = \frac{\hat{y}_{i,t}^2}{\hat{y}_{i,t}^2 + r_{i,t}^2} = \left(1 + \frac{r_{i,t}^2}{\hat{y}_{i,t}^2}\right)^{-1}, \tag{19}$$

for $\hat{y}_{i,t} \neq 0$, and the original ratio is well-defined for all coordinates. Thus $q_{i,t}$ is a monotone function of the prediction signal-to-residual ratio: it is close to one when the local DMD prediction explains the current coordinate and close to zero when the residual dominates. Therefore Eq. (9) is an observation-centric soft spectral decomposition: the DMD model determines the gates, while the routed content remains proportional to the observed update coordinate $y_{i,t}$. This avoids synthesizing a new update from the prediction and keeps `HUR` aligned with the current base prompt update.

Combining Eq. (9) with the lifted residual in Eq. (5), the prompt-space components satisfy $\mathbf{u}_t = \mathbf{S}_t + \mathbf{C}_t + \mathbf{N}_t$ as used in Eq. (13). Thus the residual branch contains both the projected unexplained component $(1 - \mathbf{q}_t)\mathbf{y}_t$ and the residual $\widetilde{\mathbf{e}}_t$. The subsequent theorem uses this spectral partition only for bounded routing; `HUR` does not use $\rho(A_t)$ to decide persistent versus oscillatory content, and `AKC` does not use the phase partition to scale magnitudes.

B Proof of Lemma 1

Since $g_t \geq 0$, scalar multiplication by g_t scales every eigenvalue of A_t by g_t. In the diagonal case used in Eq. (7), this is immediate:

$$\rho(g_t A_t) \;=\; \max_i |g_t a_{i,t}| \;=\; g_t \max_i |a_{i,t}| \;=\; g_t \rho(A_t). \tag{20}$$

By Eq. (10), $g_t \in [g_{\min}, 1]$, hence $g_t \rho(A_t) \leq \rho(A_t)$ with equality iff $g_t = 1$. From the definition of g_t, $g_t = 1$ if and only if $\rho_t \leq \bar{\rho}$. For the upper bound Eq. (11), we split ρ_t into three regimes according to Eq. (10): if $\rho_t \leq \bar{\rho}$, then $g_t = 1$, so $g_t \rho_t = \rho_t \leq \bar{\rho}$; if $\bar{\rho} < \rho_t \leq \bar{\rho}/g_{\min}$, then $\bar{\rho}/\rho_t \geq g_{\min}$, so $g_t = \bar{\rho}/\rho_t$ and $g_t \rho_t = \bar{\rho}$; if $\rho_t > \bar{\rho}/g_{\min}$, then $\bar{\rho}/\rho_t < g_{\min}$, so $g_t = g_{\min}$ and $g_t \rho_t = g_{\min} \rho_t$. Combining the three cases with Eq. (20), we obtain $\rho(g_t A_t) = g_t \rho_t \leq \max(\bar{\rho}, g_{\min} \rho_t)$, which gives Eq. (11). In particular, when $\rho_t \leq \bar{\rho}/g_{\min}$, the first two cases apply and $g_t \rho_t \leq \bar{\rho}$. □

Remark (diagonal vs. full Koopman). The spectral homogeneity in Lemma 1 does not require A_t to be diagonal. The diagonal local model is instead used to provide scalar mode coefficients $\{a_{i,t}\}_{i=1}^r$ for the gates in Eq. (8), the coordinate-wise decomposition in Eq. (9), and the mode-wise routing property in Theorem 1. The following lemma summarizes why this coordinate-wise estimator is preferable in the local Hankel regime. Let $A_t^{\text{full}} = Y_t^+ (Y_t^-)^\dagger$ be the full-matrix alternative. Under the fixed-design local linear model $\mathbf{y}_{\tau+1} = A^\star \mathbf{y}_\tau$, define $G \coloneqq Y_t^- (Y_t^-)^\top$. Since Y_t^- contains $M - 1$ transition columns and $r \leq \min(Kd, M)$, the boundary case $r \geq M$ can arise when the selected rank reaches the number of Hankel columns.

Lemma 4 (Diagonal estimator in the local Hankel regime)**.** *Under the setting above, the full Koopman estimator A_t^{full} and the diagonal estimator $A_t = \operatorname{diag}(a_{1,t}, \ldots, a_{r,t})$ satisfy:*

(i) Underdetermined full estimator. *If $r \geq M$, then $\operatorname{rank}(Y_t^-) \leq M - 1 < r$, so the least-squares minimizers for $AY_t^- \approx Y_t^+$ form an affine set with at least $r(r - M + 1)$ free dimensions. The Moore–Penrose estimator selects one minimizer, but the dynamics on $\ker((Y_t^-)^\top)$ remain unconstrained by the local data. In contrast, each diagonal coefficient in Eq. (7) is a scalar least-squares estimate and is unique for every mode satisfying $\sum_{\tau=t-M}^{t-2} y_{i,\tau}^2 > 0$.*

(ii) Local estimation variance. *If $r < M$ and $G \succ 0$, the full least-squares estimator satisfies $\mathbb{E}\|\hat{A}^{\text{full}} - A^\star\|_F^2 = r\sigma^2 \text{tr}(G^{-1})$ under the fixed-design noise model. For the mode-wise scalar estimator in Eq.* (7)*, the conditional noise variance of $\hat{a}_{i,t}$ is σ^2/G_{ii}. Thus the full estimator estimates coupled cross-coordinate effects from a short local window, whereas the diagonal estimator estimates only the scalar coefficient used by the corresponding gate.*

(iii) Compatibility with the decomposition. *The gates in Eq.* (8) *and the decomposition in Eq.* (9) *are defined for each Hankel coordinate through the scalar coefficient $a_{i,t}$. A full matrix mixes coordinates and therefore does not provide the same coordinate-wise quantities $\{a_{i,t}\}_{i=1}^r$ required to form $q_{i,t}$, $\pi_{i,t}$, and the weight $w_{i,t} \in [\beta_n, 1]$ in Theorem 1(ii).*

(iv) Per-step cost. *Computing A_t costs $\mathcal{O}(rM)$ for the diagonal estimator versus $\mathcal{O}(r^2M + r^3)$ for the full estimator.*

C Proof of Theorem 1

Let

$$\widetilde{\mathbf{e}}_t = (I_{Kd} - Q_t Q_t^\top)\mathbf{z}_t$$

be the Hankel-lifted residual from Eq. (4). The lifted routed update is

$$\widetilde{\mathbf{z}}_t^{\text{hur}} = Q_t\big(\mathbf{s}_t^\star + \beta_c \mathbf{c}_t^\star + \beta_n \mathbf{n}_t^\star\big) + \beta_n \widetilde{\mathbf{e}}_t, \qquad \mathbf{u}_t^{\text{hur}} = E_K \widetilde{\mathbf{z}}_t^{\text{hur}}. \tag{21}$$

Combining Eq. (9) with Eq. (21), the i-th routed Hankel coordinate satisfies

$$s_{i,t}^\star + \beta_c c_{i,t}^\star + \beta_n n_{i,t}^\star = \big[q_{i,t}\pi_{i,t} + \beta_c q_{i,t}(1-\pi_{i,t}) + \beta_n(1-q_{i,t})\big] y_{i,t} =: w_{i,t} y_{i,t}. \tag{22}$$

Since $q_{i,t}, \pi_{i,t} \in [0,1]$ and $0 \le \beta_n \le \beta_c \le 1$, and since $q_{i,t}\pi_{i,t} + q_{i,t}(1-\pi_{i,t}) + (1-q_{i,t}) = 1$, the coefficient $w_{i,t}$ is a convex combination of 1, β_c, and β_n. Therefore,

$$w_{i,t} \in [\beta_n, 1]. \tag{23}$$

Proof of (i). Because Q_t has orthonormal columns and $\widetilde{\mathbf{e}}_t \perp \text{range}(Q_t)$ in the Hankel-lifted space,

$$\|\widetilde{\mathbf{z}}_t^{\text{hur}}\|^2 = \sum_{i=1}^r w_{i,t}^2 y_{i,t}^2 + \beta_n^2 \|\widetilde{\mathbf{e}}_t\|^2 \le \sum_{i=1}^r y_{i,t}^2 + \|\widetilde{\mathbf{e}}_t\|^2 = \|\mathbf{z}_t\|^2, \tag{24}$$

where we used $w_{i,t} \le 1$ and $\beta_n \le 1$. Since $\|E_K\|_2 = 1$ and $g_t \in (0,1]$, we have

$$\|\Delta \mathbf{p}_t\| = g_t \|\mathbf{u}_t^{\text{hur}}\| = g_t \|E_K \widetilde{\mathbf{z}}_t^{\text{hur}}\| \le g_t \|\widetilde{\mathbf{z}}_t^{\text{hur}}\| \le g_t \|\mathbf{z}_t\| \le \|\mathbf{z}_t\|. \tag{25}$$

This proves the Hankel-space non-amplification statement and the induced prompt-update bound.

Proof of (ii). From Eq. (22), the projected Hankel coordinate after routing equals $w_{i,t} y_{i,t}$. Since $w_{i,t} \ge \beta_n \ge 0$, the sign of the projected coordinate is not reversed; since $w_{i,t} \le 1$, its magnitude is not amplified.

Proof of (iii). The lifted persistent component $\widetilde{\mathbf{S}}_t$ is transmitted with coefficient 1, while $\widetilde{\mathbf{C}}_t$ and $\widetilde{\mathbf{N}}_t$ are attenuated by factors β_c and β_n, respectively. Applying the linear extraction E_K gives the same coefficients for their prompt-space extractions $\mathbf{S}_t$, $\mathbf{C}_t$, and $\mathbf{N}_t$ in Eq. (14). □

D Full Experiments

Together with the main tables, these appendix results form a multi-axis stress test targeted at the core claim of TPD: prompt-agnostic stabilization of update trajectories rather than dataset- or baseline-specific tuning. The evaluation spans datasets, prompt modalities, backbone families, adaptation modes, distribution-shift protocols, TPT baselines, simple stabilization rules, and component ablations, making the improvements difficult to explain by a narrow choice of dataset, prompt type, backbone, or heuristic stabilizer.

D.1 Full Experimental Details

Datasets. Following previous methods [Shu et al., 2022], we evaluate our method on 15 datasets under two settings that suffer from distribution shifts: domain generalization and cross-dataset transfer. For the domain generalization setting, we evaluate on ImageNet [Deng et al., 2009] and its four OOD variants: ImageNet-V2 [Recht et al., 2019], ImageNet-Sketch [Wang et al., 2019], ImageNet-A [Hendrycks et al., 2021b], and ImageNet-R [Hendrycks et al., 2021a]. For the cross-dataset setting, we evaluate on 10 datasets covering diverse tasks: Caltech101 [Fei-Fei et al., 2004], OxfordPets [Parkhi et al., 2012], StanfordCars [Krause et al., 2013], Flowers102 [Nilsback and Zisserman, 2008], Food101 [Bossard et al., 2014], FGVCAircraft [Maji et al., 2013], SUN397 [Xiao et al., 2010], DTD [Cimpoi et al., 2014], EuroSAT [Helber et al., 2019], and UCF101 [Soomro et al., 2012].

Baseline Models. We attach `TPD` to state-of-the-art TPT methods on two backbone families. On the ViT side [Dosovitskiy et al., 2020], we integrate `TPD` with VPT [Jia et al., 2022], E2VPT [Han et al., 2023], VFPT [Zeng et al., 2024], and BPT [Wang et al., 2025b], all using a supervised ViT-B/16 pretrained on ImageNet-21k [Deng et al., 2009]. On the CLIP side [Radford et al., 2021], we evaluate representative text TPT baselines TPT [Shu et al., 2022], C-TPT [Yoon et al., 2024], R-TPT [Sheng et al., 2025], O-TPT [Sharifdeen et al., 2025], CAT [Zhang et al., 2025] and DyPt [Xiao et al., 2025] in both episodic and online modes. `TPD` maintains only update-history statistics across samples and does not carry adapted prompt parameters across episodes. For each baseline, we report both the original result and its `TPD`-augmented variant, denoted as "Baseline +`TPD`".

Implementation Details. We fix `TPD` hyperparameters across backbones. The local Hankel subspace uses a delay length of $K{=}3$ and $M{=}8$ Hankel columns, corresponding to a local window of $W{=}K+M-1{=}10$ past prompt updates. The effective rank r is determined adaptively from the singular spectrum and satisfies $r \leq \min(Kd, M)$. Before W past updates are available, we use the base prompt update without `TPD` control. For `AKC`, we set the stability threshold $\bar{\rho}{=}1.0$ and the minimum gain $g_{\min}{=}0.1$. For `HUR`, we use $\beta_c{=}0.25$, $\beta_n{=}0.1$ on ViT and CLIP.

On the ViT branch, we follow the VPT [Jia et al., 2022] protocol with 10 deep prompt tokens. We perform one TTA step per test sample with learning rate 10^{-3} using SGD, a batch size of 128, and select the top 10% most confident samples for the entropy-minimization objective [Wang et al., 2020]. On the CLIP branch, we build on CoOp [Zhou et al., 2022] with $N{=}4$ learnable context tokens initialized from "`a photo of a`". In episodic mode, we follow the standard episodic protocol by resetting the prompt state after each episode. The only cross-episode state maintained by `TPD` is a lightweight history of recent prompt-update vectors used by the controller to estimate local trajectory stability. Importantly, `TPD` does not carry adapted prompt parameters, optimizer states, or episode-specific prompt features across episodes. This separates model adaptation from trajectory stabilization: each episode starts from the same reset prompt state as the base episodic method, while the controller uses past update statistics only to decide how to scale and route the current prompt update. All experiments are run on an NVIDIA H100 GPU.

D.2 Full Per-Dataset Results

We provide full per-dataset results for the compressed comparisons and ablations reported in the main paper. Tables A1–A4 expand the simple stabilization comparison in Table 3, covering update clipping, EMA update, norm control, low-pass filtering, and `TPD` on VPT and TPT under DG and CSD settings. Tables A5–A10 expand the ablation study in Table 4, separately reporting Koopman operator choices, projection choices, and `HUR` routing variants. These tables show that the mean improvements in the main paper are supported by broad per-dataset gains rather than by a small number of outlier datasets.

Furthermore, VPT, E2VPT, and VFPT were originally developed as training-time visual prompt tuning methods rather than test-time adaptation algorithms. In our setting, we adapt them with the same entropy-minimization objective used in standard TPT, but this naive test-time variant has no mechanism for stabilizing prompt-update trajectories. As a result, visual-prompt updates can expand, oscillate, or drift under unlabeled target samples, and the resulting OOD mean may fall below the frozen ViT zero-shot baseline, e.g., 39.66% versus 46.17% in Table 1. This does not indicate that visual prompt tuning itself is ineffective; rather, it shows that naive TTA on visual prompts remains unstable without trajectory-level control. `TPD` addresses this missing stabilizer by regulating update

magnitude and refining update direction, turning unstable test-time tuning into effective adaptation that surpasses zero-shot under shifted-domain evaluation and remains competitive under cross-dataset transfer.

D.2.1 Simple stabilization baselines.

The simple stabilization baselines improve over the raw TPT updates on several datasets, but their gains are smaller and less consistent than TPD. In particular, clipping and norm control mainly restrict update scale, while EMA and low-pass filtering smooth the trajectory without distinguishing persistent, oscillatory, and residual components. The per-dataset results below show that the trajectory-aware decomposition and routing in TPD provide more reliable improvements across both visual and text prompts.

Table A1: Per-dataset simple stabilization results for VPT under domain generalization. OoD Mean_4 averages ImageNet-V2, ImageNet-Sketch, ImageNet-A, and ImageNet-R.

Variant	ImageNet	ImageNet-V2	ImageNet-Sketch	ImageNet-A	ImageNet-R	OoD Mean_4
Baseline	78.16	64.55	32.77	19.17	42.14	39.66
+ Update clipping	78.84	65.52	34.09	22.95	45.11	41.92
+ EMA update	79.35	65.98	36.82	24.44	43.71	42.74
+ Norm control	79.76	66.66	34.26	25.60	48.09	43.65
+ Low-pass filtering	80.14	68.29	34.76	28.43	45.65	44.28
VPT+TPD	81.31	69.87	36.02	38.98	46.47	**47.84**

Table A2: Per-dataset simple stabilization results for VPT under cross-dataset transfer. Mean averages the ten downstream datasets.

Variant	Caltech	Pets	Cars	Flowers	Food	Aircraft	SUN	DTD	EuroSAT	UCF	Mean
Baseline	92.65	86.89	60.68	64.86	85.14	21.21	59.92	38.19	20.91	59.28	58.97
+ Update clipping	92.94	87.35	61.27	65.16	85.85	21.93	60.57	38.48	21.78	60.03	59.54
+ EMA update	93.27	87.46	61.07	65.33	85.67	22.47	61.00	38.77	22.79	60.80	59.86
+ Norm control	93.10	87.71	63.33	66.75	86.59	26.40	60.40	40.54	24.25	61.19	61.03
+ Low-pass filtering	93.39	89.29	62.23	67.32	86.20	21.99	62.55	39.07	25.15	61.93	60.91
VPT+TPD	95.37	89.38	62.81	68.09	86.45	24.46	63.79	42.25	32.17	62.01	**62.68**

Table A3: Per-dataset simple stabilization results for TPT under domain generalization. OOD Mean averages the four shifted variants excluding ImageNet.

Variant	ImageNet	ImageNet-V2	ImageNet-Sketch	ImageNet-A	ImageNet-R	OOD Mean
Baseline	68.98	63.45	47.94	54.77	77.06	60.81
+ Update clipping	69.90	64.12	48.40	55.09	77.84	61.36
+ EMA update	69.86	63.80	49.82	55.25	77.74	61.65
+ Norm control	70.11	63.64	50.05	55.80	77.39	61.72
+ Low-pass filtering	70.62	64.49	49.02	56.03	79.38	62.23
TPT+TPD	71.19	65.98	49.62	56.73	79.98	**63.08**

Table A4: Per-dataset simple stabilization results for TPT under cross-dataset transfer. Mean averages the ten downstream datasets.

Variant	Caltech	Pets	Cars	Flowers	Food	Aircraft	SUN	DTD	EuroSAT	UCF	Mean
Baseline	94.16	87.79	66.87	68.98	84.67	24.78	65.50	47.75	42.44	68.04	65.10
+ Update clipping	94.48	88.06	67.12	69.63	84.88	25.17	65.59	48.52	42.78	68.49	65.47
+ EMA update	94.24	88.53	67.47	69.75	85.13	26.04	65.91	48.58	43.85	68.73	65.82
+ Norm control	94.60	88.77	67.84	68.98	85.26	25.80	65.77	49.00	42.93	68.49	65.74
+ Low-pass filtering	94.56	88.66	68.09	69.83	85.58	26.34	66.37	51.18	43.11	68.57	66.23
TPT+TPD	96.32	90.31	68.55	71.06	88.32	27.19	68.51	51.98	46.03	72.00	**68.03**

D.2.2 Full detailed ablations.

We next report per-dataset ablations for the main components of TPD. The Koopman ablation compares full and diagonal local estimators; the projection ablation compares plain projection with the Hankel construction; and the routing ablation tests whether preserving persistent components while attenuating oscillatory and residual components is beneficial. Across datasets, the diagonal estimator, Hankel projection, and full routing rule give the most stable overall behavior.

For episodic TPT, prompts are reset after each episode. TPD does not carry adapted prompt parameters across episodes; it only maintains lightweight update-history statistics for trajectory analysis. If these update-history statistics are ablated and reset after every episode as well, no valid cross-step trajectory is available. In this case, TPD applies identity control, *i.e.*, $g_t = 1$ and $\mathbf{u}_t^{\mathrm{hur}} = \mathbf{u}_t$, so the resulting method is exactly the corresponding episodic baseline. Therefore, this ablation is already represented by the baseline rows.

Table A5: Per-dataset Koopman operator ablation for VPT. **(a)** Domain generalization.

Variant	ImageNet	ImageNet-V2	ImageNet-Sketch	ImageNet-A	ImageNet-R	OoD_4
Baseline	78.16	64.55	32.77	19.17	42.14	39.66
Full Koopman	79.74	69.39	35.83	23.73	45.36	43.58
VPT+AKC (diag.)	80.94	69.70	38.33	28.47	47.17	45.92
VPT+TPD	81.31	69.87	36.02	38.98	46.47	**47.84**

Table A5: Per-dataset Koopman operator ablation for VPT. **(b)** Cross-dataset transfer.

Variant	Caltech	Pets	Cars	Flowers	Food	Aircraft	SUN	DTD	EuroSAT	UCF	Mean
Baseline	92.65	86.89	60.68	64.86	85.14	21.21	59.92	38.19	20.91	59.28	58.97
Full Koopman	93.67	88.14	61.30	66.34	85.88	24.63	62.44	39.78	24.84	61.56	60.86
VPT+AKC (diag.)	94.24	89.78	62.33	66.83	86.03	25.23	62.15	40.96	26.95	62.94	61.74
VPT+TPD	95.37	89.38	62.81	68.09	86.45	24.46	63.79	42.25	32.17	62.01	**62.68**

Table A6: Per-dataset Koopman operator ablation for TPT. **(a)** Domain generalization.

Variant	ImageNet	ImageNet-V2	ImageNet-Sketch	ImageNet-A	ImageNet-R	OOD Mean
Baseline	68.98	63.45	47.94	54.77	77.06	60.81
Full Koopman	69.86	63.73	49.87	55.83	77.76	61.80
TPT+AKC (diag.)	71.28	64.36	50.52	56.28	79.09	62.56
TPT+TPD	71.19	65.98	49.62	56.73	79.98	**63.08**

Table A6: Per-dataset Koopman operator ablation for TPT. **(b)** Cross-dataset transfer.

Variant	Caltech	Pets	Cars	Flowers	Food	Aircraft	SUN	DTD	EuroSAT	UCF	Mean
Baseline	94.16	87.79	66.87	68.98	84.67	24.78	65.50	47.75	42.44	68.04	65.10
Full Koopman	95.29	88.91	68.05	70.44	85.86	26.52	66.46	48.17	44.65	69.47	66.38
TPT+AKC (diag.)	95.42	89.18	68.77	70.60	88.29	26.43	68.19	50.30	44.95	69.97	67.21
TPT+TPD	96.32	90.31	68.55	71.06	88.32	27.19	68.51	51.98	46.03	72.00	**68.03**

Table A7: Per-dataset projection ablation for VPT. **(a)** Domain generalization.

Variant	ImageNet	ImageNet-V2	ImageNet-Sketch	ImageNet-A	ImageNet-R	OoD_4
Baseline	78.16	64.55	32.77	19.17	42.14	39.66
Plain projection	79.17	67.46	34.53	23.41	43.83	42.31
VPT+HUR (Hankel)	81.10	69.84	38.60	26.25	49.21	45.98
VPT+TPD	81.31	69.87	36.02	38.98	46.47	**47.84**

Table A7: Per-dataset projection ablation for VPT. **(b)** Cross-dataset transfer.

Variant	Caltech	Pets	Cars	Flowers	Food	Aircraft	SUN	DTD	EuroSAT	UCF	Mean
Baseline	92.65	86.89	60.68	64.86	85.14	21.21	59.92	38.19	20.91	59.28	58.97
Plain projection	93.87	86.92	61.57	65.41	85.84	23.01	61.29	39.95	24.33	60.03	60.22
VPT+HUR (Hankel)	95.21	89.26	64.99	67.07	86.85	25.26	64.69	40.37	26.81	61.30	62.18
VPT+TPD	95.37	89.38	62.81	68.09	86.45	24.46	63.79	42.25	32.17	62.01	**62.68**

Table A8: Per-dataset projection ablation for TPT. **(a)** Domain generalization.

Variant	ImageNet	ImageNet-V2	ImageNet-Sketch	ImageNet-A	ImageNet-R	OOD Mean
Baseline	68.98	63.45	47.94	54.77	77.06	60.81
Plain projection	69.46	64.31	48.65	54.99	77.78	61.43
TPT+HUR (Hankel)	70.77	65.31	50.24	56.77	79.05	62.84
TPT+TPD	71.19	65.98	49.62	56.73	79.98	**63.08**

Table A8: Per-dataset projection ablation for TPT. **(b)** Cross-dataset transfer.

Variant	Caltech	Pets	Cars	Flowers	Food	Aircraft	SUN	DTD	EuroSAT	UCF	Mean
Baseline	94.16	87.79	66.87	68.98	84.67	24.78	65.50	47.75	42.44	68.04	65.10
Plain projection	94.77	88.17	67.42	70.20	85.43	26.04	66.19	48.76	43.56	69.52	66.01
TPT+HUR (Hankel)	95.25	89.07	67.98	71.74	86.68	27.18	68.60	53.01	45.59	70.34	67.54
TPT+TPD	96.32	90.31	68.55	71.06	88.32	27.19	68.51	51.98	46.03	72.00	**68.03**

Table A9: Per-dataset HUR routing ablation for VPT. **(a)** Domain generalization.

Variant	ImageNet	ImageNet-V2	ImageNet-Sketch	ImageNet-A	ImageNet-R	OoD_4
Baseline	78.16	64.55	32.77	19.17	42.14	39.66
w/o Pres.	81.33	69.32	35.99	36.51	46.41	47.06
w/o Osc.	81.36	69.38	36.48	36.65	46.61	47.28
w/o Res.	81.02	69.08	35.25	36.35	46.68	46.84
VPT+TPD	81.31	69.87	36.02	38.98	46.47	**47.84**

Table A9: Per-dataset HUR routing ablation for VPT. **(b)** Cross-dataset transfer.

Variant	Caltech	Pets	Cars	Flowers	Food	Aircraft	SUN	DTD	EuroSAT	UCF	Mean
Baseline	92.65	86.89	60.68	64.86	85.14	21.21	59.92	38.19	20.91	59.28	58.97
w/o Pres.	95.42	88.96	62.50	67.97	85.98	24.54	63.06	41.19	31.81	61.49	62.29
w/o Osc.	95.66	88.09	62.19	68.21	86.34	24.00	61.89	41.78	33.26	62.28	62.37
w/o Res.	94.69	88.96	62.16	67.84	86.01	23.85	63.08	43.03	30.81	61.64	62.21
VPT+TPD	95.37	89.38	62.81	68.09	86.45	24.46	63.79	42.25	32.17	62.01	**62.68**

Table A10: Per-dataset HUR routing ablation for TPT. **(a)** Domain generalization.

Variant	ImageNet	ImageNet-V2	ImageNet-Sketch	ImageNet-A	ImageNet-R	OOD Mean
Baseline	68.98	63.45	47.94	54.77	77.06	60.81
w/o Pres.	70.54	65.86	49.05	56.71	80.17	62.95
w/o Osc.	71.25	65.58	49.42	56.36	80.00	62.84
w/o Res.	71.70	66.02	49.22	56.36	79.61	62.80
TPT+TPD	71.19	65.98	49.62	56.73	79.98	**63.08**

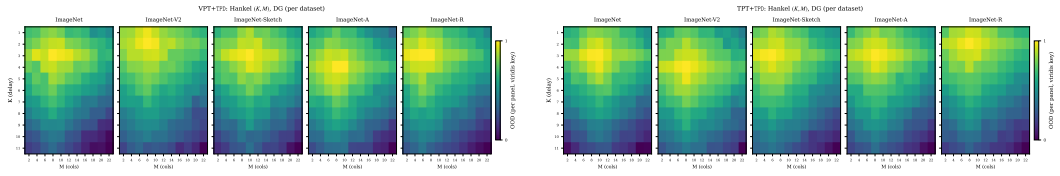
Table A10: Per-dataset HUR routing ablation for TPT. **(b)** Cross-dataset transfer.

Variant	Caltech	Pets	Cars	Flowers	Food	Aircraft	SUN	DTD	EuroSAT	UCF	Mean
Baseline	94.16	87.79	66.87	68.98	84.67	24.78	65.50	47.75	42.44	68.04	65.10
w/o Pres.	96.27	90.35	68.00	70.65	88.08	27.06	68.88	51.65	45.48	70.90	67.73
w/o Osc.	96.92	89.89	68.10	70.36	89.18	26.16	68.99	52.01	46.30	70.82	67.87
w/o Res.	96.47	89.62	68.19	70.24	87.38	27.48	67.96	51.24	46.06	70.79	67.54
TPT+TPD	96.32	90.31	68.55	71.06	88.32	27.19	68.51	51.98	46.03	72.00	**68.03**

D.3 Hyperparameter Sensitivity

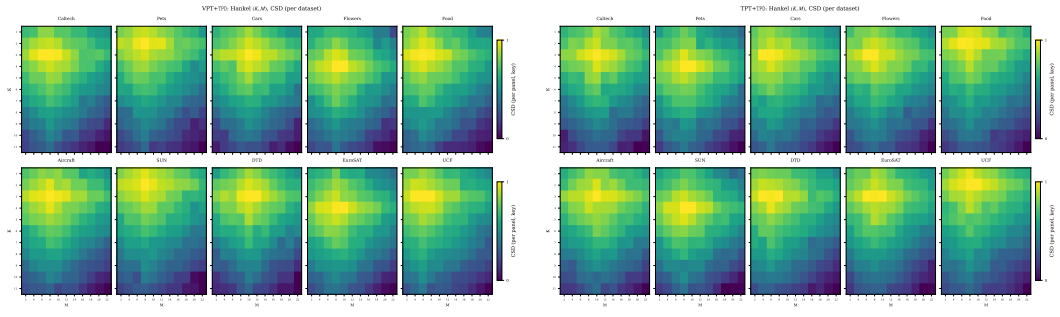
We provide per-dataset sensitivity analyses for the three groups of TPD hyperparameters: the Hankel window (K, M), the AKC control parameters $(\bar{\rho}, g_{\min})$, and the HUR routing weights (β_c, β_n). All heatmaps are normalized within each panel for visualization, so warmer colors indicate better performance for the same dataset and baseline. Overall, the high-performing regions are broad rather than isolated, showing that TPD is not sensitive to a single carefully tuned setting. All main-table results use a single fixed TPD configuration across datasets, prompt modalities, backbone families, and baselines. The sensitivity sweeps in this section are diagnostic analyses for robustness, not per-dataset or per-baseline hyperparameter selection. Thus, the reported gains do not rely on choosing target-specific controller parameters.

Hankel window. We first sweep the Hankel delay length K and the number of Hankel columns M. Across both VPT and TPT, very short histories provide insufficient temporal context, while overly long histories may mix outdated updates into the local model. In contrast, short-to-moderate windows consistently form a stable high-performing region across both DG and CSD datasets. The default setting $K = 3$ and $M = 8$ lies inside this region, supporting its use as a fixed configuration across prompt types and datasets.



(a) VPT+TPD, domain generalization (b) TPT+TPD, domain generalization

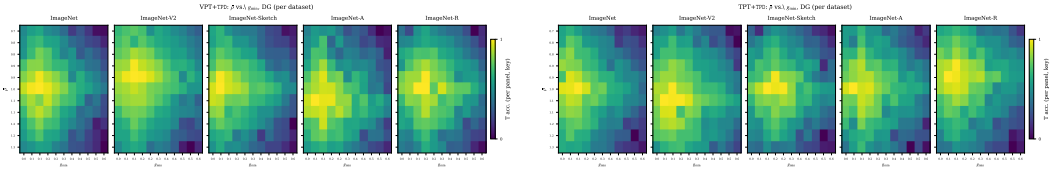
Figure 6: Sensitivity to the Hankel window on domain-generalization datasets. Each heatmap reports per-dataset performance of VPT+TPD or TPT+TPD while sweeping the delay length K and the number of Hankel columns M. Colors are normalized within each panel, with warmer colors indicating higher accuracy. Both prompt types show a broad high-performing region at short-to-moderate windows, supporting the default $K = 3, M = 8$.



(a) VPT+TPD, CSD (b) TPT+TPD, CSD

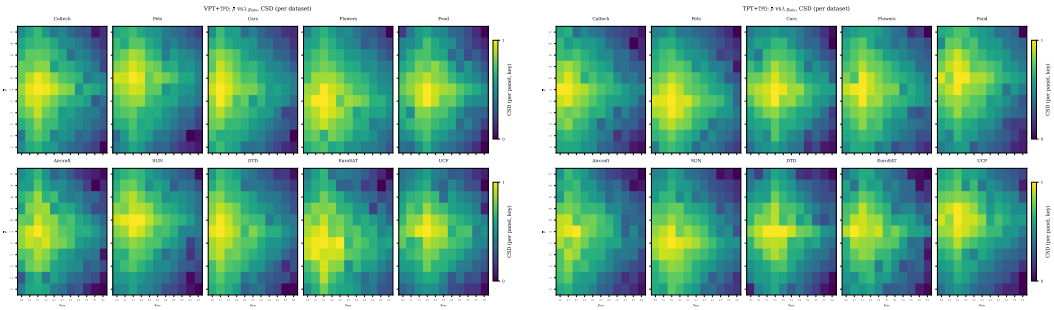
Figure 7: Sensitivity to the Hankel window on cross-dataset transfer. The same (K, M) sweep is evaluated on ten downstream datasets. The high-performing region remains broad across datasets and prompt types, while excessively short or long histories are less reliable.

AKC threshold and gain floor. We next sweep the AKC stability threshold $\bar{\rho}$ and the minimum gain $g_{\min}$. The results show a consistent plateau around $\bar{\rho} = 1.0$ with a small gain floor. This matches the intended role of AKC: the controller should intervene when the local Koopman dynamics become expansive, but it should not over-dampen stable updates. Large $g_{\min}$ values weaken this intervention by preventing sufficient shrinkage, while overly strict thresholds can suppress useful updates. The default $\bar{\rho} = 1.0$ and $g_{\min} = 0.1$ balance these two effects.



(a) VPT+TPD, domain generalization (b) TPT+TPD, domain generalization

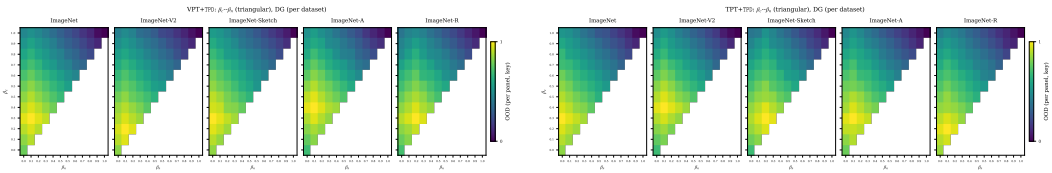
Figure 8: Sensitivity of AKC on domain-generalization datasets. Each heatmap sweeps the spectral-radius threshold $\bar{\rho}$ and gain floor $g_{\min}$ for VPT+TPD or TPT+TPD. Colors are normalized within each panel. Performance is strongest in a broad region around $\bar{\rho} = 1.0$ with a small gain floor, indicating that AKC benefits from suppressing locally expansive dynamics without over-damping stable updates.



(a) VPT+TPD, CSD (b) TPT+TPD, CSD

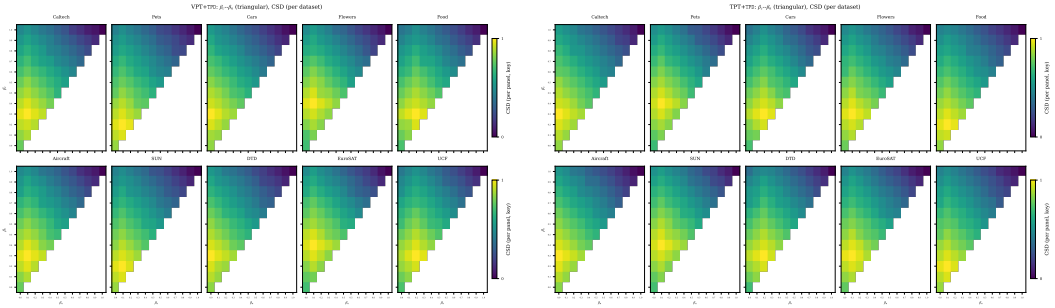
Figure 9: Sensitivity of AKC on cross-dataset transfer. The $\bar{\rho}$–$g_{\min}$ sweep shows that moderate stability control is robust across datasets, whereas a large gain floor weakens the controller and overly strict settings may suppress useful adaptation.

HUR routing weights. Finally, we sweep the HUR routing weights under the constraint $0 \leq \beta_n \leq \beta_c \leq 1$. The best regions favor strong suppression of residual components and moderate attenuation of oscillatory components. This agrees with the design of HUR: persistent components are preserved with unit weight, oscillatory components are attenuated rather than discarded, and poorly explained residual components are suppressed more strongly. Performance degrades when both weights approach one, where HUR becomes close to the raw update and loses its selective-routing effect.



(a) VPT+TPD, domain generalization (b) TPT+TPD, domain generalization

Figure 10: Sensitivity of HUR routing weights on domain-generalization datasets. Each heatmap sweeps the triangular grid $0 \leq \beta_n \leq \beta_c \leq 1$, where β_c controls the oscillatory component and β_n controls the residual component; the masked region has $\beta_n > \beta_c$. Warmer colors indicate higher per-panel accuracy. The high-performing region favors small residual weights and moderate oscillatory attenuation, supporting selective routing rather than using the raw update unchanged.



(a) VPT+TPD, CSD (b) TPT+TPD, CSD

Figure 11: Sensitivity of HUR routing weights on cross-dataset transfer. The triangular β_c–β_n sweep is consistent across VPT+TPD and TPT+TPD: retaining persistent components while attenuating oscillatory and residual components yields a broad high-performing region, whereas weights near one reduce HUR to weak routing and degrade performance.

D.4 Persistent Ratio and Adaptation Gain

We further examine the link between persistent ratio and adaptation gain. The persistent ratio measures the fraction of a prompt update aligned with locally consistent directions in the recent update trajectory. A larger value indicates more stable adaptation content and less oscillatory or residual content.

For each dataset, we plot the persistent ratio against the accuracy gain obtained by TPD. The red dashed curve is a piecewise linear fitted trend line over the persistent-ratio axis. Across TPT, the gain generally increases from low to moderate persistent ratios and reaches a peak at an intermediate range, after which the trend saturates or slightly decreases on several datasets. This pattern suggests that persistent ratio is a useful trajectory-level diagnostic, while excessively persistent updates may no longer bring additional adaptation benefit.

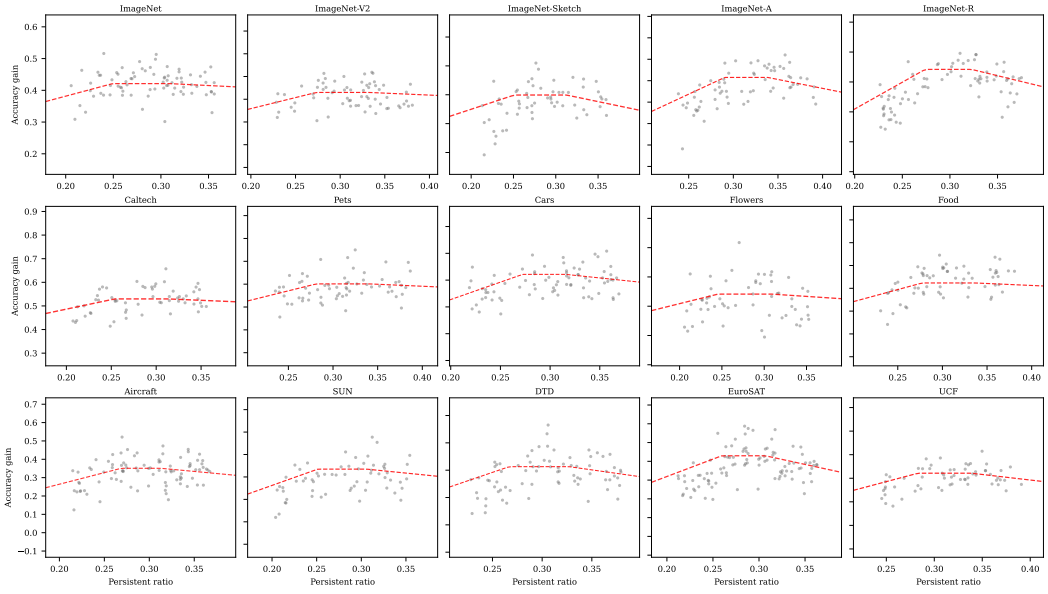


Figure 12: Persistent ratio versus accuracy gain for TPT+TPD. Each point corresponds to a local trajectory segment, and the red dashed line shows the fitted trend for each dataset.

D.5 Nonstationary Streams and Test-Order Variation

We evaluate whether recent update history remains useful under abrupt, mixed, and rapidly alternating domain shifts. Each stream is evaluated using five paired test orders shared by VPT and VPT+TPD.

Table A11 shows that TPD improves overall accuracy by 1.42, 1.53, and 4.44 percentage points, respectively, while reducing the standard deviation across orders in all three settings. Under Rapid Alternation, First-10 accuracy also increases from 45.82% to 50.09%, indicating improved adaptation immediately after domain switches. These results suggest that finite-window history remains informative under the tested distribution changes, including a domain switch after every update.

Table A11: Accuracy (%) under nonstationary streams. Overall reports mean $\pm$ standard deviation across five paired test orders; the other accuracy columns report the corresponding means.

Stream	Method	Overall	V2	Sketch	First-10
Abrupt	VPT	52.90 ± 0.37	64.52	29.67	47.28
	VPT+TPD	54.32 ± 0.32	65.78	31.40	48.78
Mixed	VPT	46.10 ± 0.25	65.09	27.11	N/A
	VPT+TPD	47.63 ± 0.19	66.19	29.08	N/A
Rapid Alternation	VPT	45.22 ± 0.51	63.82	26.62	45.82
	VPT+TPD	49.66 ± 0.27	67.64	31.69	50.09

Setup. Let V and K denote ImageNet-V2 and ImageNet-Sketch. Abrupt: $V_{5000} \to K_{5000} \to V_{5000}$ (15,000 samples). Mixed: a global shuffle of 10,000 samples from each domain (20,000 samples). Rapid Alternation: $V_{128} \to K_{128} \to \cdots$ (20,000 samples), with a domain switch after every adaptation update. VPT and VPT+TPD use the same five predefined test orders for each stream. First-10 measures accuracy over the first ten batches after a domain switch and is undefined for Mixed.

D.6 Component Ratios under Nonstationary Shifts

Table A12 examines how the decomposition changes under nonstationary streams. For VPT, the persistent fraction decreases from 17.54% under Abrupt to 5.09% under Rapid Alternation, while the residual fraction increases from 81.75% to 93.65%. Nevertheless, TPD improves accuracy in every stream, with its largest VPT gain under Rapid Alternation. TPT also benefits despite residual fractions exceeding 96%. These ratios characterize local trajectory predictability rather than semantic correctness. The results support soft attenuation with nonzero routing weights, which allows potentially useful corrections to remain active when updates are weakly predictable.

Table A12: Component ratios and accuracy (%) under nonstationary shifts. P, O, and R denote persistent, oscillatory, and residual fractions. Accuracy is reported as Base $\to$ Base+TPD.

Method	Stream	P	O	R	Accuracy
VPT	Abrupt	17.54	0.71	81.75	$52.90 \to 54.32$
VPT	Mixed	12.83	0.48	86.69	$46.10 \to 47.63$
VPT	Rapid Alternation	5.09	1.26	93.65	$45.22 \to 49.66$
TPT	Abrupt	3.30	0.45	96.25	$57.31 \to 57.92$
TPT	Mixed	3.16	0.44	96.40	$54.85 \to 55.58$
TPT	Rapid Alternation	3.19	0.46	96.35	$55.09 \to 55.82$

Note. Streams follow Table A11. The component ratios describe local trajectory predictability under the fitted model rather than the semantic correctness of the updates.

D.7 Variation across Random Seeds

We repeat VPT and TPT on ImageNet-A/R using seeds 4827, 7316, and 9054. Table A13 reports the mean and standard deviation across these independent runs. TPD improves mean accuracy in all four settings and reduces the standard deviation in three. For VPT on ImageNet-A, accuracy increases from $19.25 \pm 1.97\%$ to $38.73 \pm 1.28\%$. For TPT on ImageNet-A, the standard deviation increases slightly from 1.12 to 1.16, showing that improved mean accuracy does not uniformly imply lower run-to-run variation.

Table A13: Accuracy (%) as mean ± standard deviation across three random seeds (4827, 7316, and 9054). Base denotes the corresponding VPT or TPT method without TPD.

	VPT		TPT	
Method	ImageNet-A	ImageNet-R	ImageNet-A	ImageNet-R
Base	19.25 ± 1.97	42.32 ± 0.61	54.44 ± 1.12	77.01 ± 2.05
Base+TPD	38.73 ± 1.28	46.54 ± 0.54	56.62 ± 1.16	79.63 ± 1.89

D.8 Comparison with Adaptive Magnitude Controllers

We compare TPD with Adaptive Clipping and Variance-aware Scaling, two label-free magnitude controllers using the same base updates and evaluation streams. Table A14 shows that TPD achieves the highest accuracy in all four settings. For VPT on ImageNet-A, TPD reaches 38.98%, compared with 23.52% for Adaptive Clipping and 22.76% for Variance-aware Scaling. These comparisons complement the component ablations by showing that the tested magnitude-only controllers do not fully account for the gains obtained through joint magnitude control and directional routing.

Table A14: Accuracy (%) of adaptive magnitude controllers using the same base updates and evaluation streams. Base denotes the corresponding VPT or TPT method without a controller.

	VPT		TPT	
Controller	ImageNet-A	ImageNet-R	ImageNet-A	ImageNet-R
Base	19.17	42.14	54.77	77.06
Adaptive Clipping	23.52	41.92	54.85	77.31
Variance-aware Scaling	22.76	43.65	55.19	77.26
TPD	38.98	46.47	56.73	79.98

Controllers. Both comparison controllers are label-free. Adaptive Clipping applies an adaptive magnitude cap. Variance-aware Scaling maps the running variance of recent update norms to a multiplicative gain.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state claims consistent with theory and experiments.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The conclusion discusses limitations.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The method and appendix provide assumptions, lemmas, theorem, and proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The experiments and appendix specify datasets, baselines, hyperparameters, and protocols.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code is available at https://github.com/reeive/TPD.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setup details tasks, datasets, settings, baselines, and hyperparameters.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The appendix reports extensive per-dataset, sensitivity, and ablation analyses supporting robustness across settings.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports H100 GPU use and wall-clock overhead.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The experiments use public benchmarks and comply with NeurIPS ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The paper focuses on methodology and does not separately discuss broader impacts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All used assets (public datasets, algorithm implementations) are cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code is available at https://github.com/reeive/TPD.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method and experiments do not use LLMs.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.