Fast-Slow Test-Time Adaptation for Online Vision-and-Language Navigation

Junyu Gao [1 2] **Xuan Yao** [1 2] **Changsheng Xu** [1 2 3]

Abstract

The ability to accurately comprehend natural language instructions and navigate to the target location is essential for an embodied agent. Such agents are typically required to execute user instructions in an online manner, leading us to explore the use of unlabeled test samples for effective online model adaptation. However, for online Vision-and-Language Navigation (VLN), due to the intrinsic nature of inter-sample online instruction execution and intra-sample multi-step action decision, frequent updates can result in drastic changes in model parameters, while occasional updates can make the model ill-equipped to handle dynamically changing environments. Therefore, we propose a Fast-Slow Test-Time Adaptation (FSTTA) approach for online VLN by performing joint decomposition-accumulation analysis for both gradients and parameters in a unified framework. Extensive experiments show that our method obtains impressive performance gains on four popular benchmarks. Code is available at https://github.com/Feliciaxyao/ICML2024-FSTTA.

1. Introduction

Developing intelligent agents capable of adhering to human directives remains a significant challenge in embodied AI. Recently, Vision-and-Language Navigation (VLN) (Anderson et al., 2018; Qi et al., 2020; Chen et al., 2022d; Yang et al., 2023), which requires an agent to comprehend natural language instructions and subsequently perform proper

[1]State Key Laboratory of Multimodal Artificial Intelligence Systems (MAIS), Institute of Automation, Chinese Academy of Sciences (CASIA) [2]School of Artificial Intelligence, University of Chinese Academy of Sciences (UCAS) [3]Peng Cheng Laboratory, ShenZhen, China. Correspondence to: Junyu Gao, Changsheng Xu <{junyu.gao, csxu}@nlpr.ia.ac.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

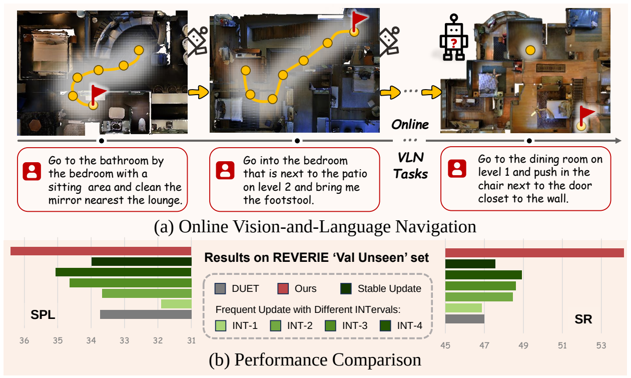


Figure 1. (a) Illustration of online VLN. (b) Comparison between TTA strategies on REVERIE (Qi et al., 2020) validation unseen set using SPL and SR metrics. 'DUET' (Chen et al., 2022d) is the base model, 'Frequent Update' means updating at certain intervals within each sample, 'Stable Update' refers to initializing with the original base model for each sample and using its best intra-sample update interval INT=1. All these strategies adopt TENT (Wang et al., 2021) for model updates. The results show that overly fast or overly slow TTA fail to achieve significant improvements.

actions to navigate to the target location, serves as a useful platform for examining the instruction-following ability.

In practical applications, a trained VLN agent is required to execute user instructions in various environments at different times in an online manner, as depicted in Figure 1(a). However, owing to disparities in environmental factors, such as distinct room types and objects, the trained agent inevitably confront data discrepancies during online testing (Gu et al., 2022; Guhur et al., 2021). This raises an important question: *can an agent accumulate experience and enhance its capabilities while executing instructions?* However, due to the lack of annotation, updating the model via supervised online learning is impractical. In addition, other learning paradigms like unsupervised domain adaptation or semi-supervised learning are also infeasible, given considerations for the issues of execution efficiency and privacy protection.

Recently, online Test-Time Adaptation (TTA) (Liang et al., 2023; Niu et al., 2023; Wang et al., 2022a) has been recognized as an effective technique of online model updating by leveraging unlabeled test samples. Although prevailing TTA methods can be integrated into VLN models with certain alterations, this direct application cannot well handle the *adaptability-stability dilemma* of models due to the intrinsic nature of inter-sample online instruction execution and intra-sample multi-step action decision. Specifically, unlike traditional classification tasks where a single TTA operation is sufficient for each test sample, as illustrated in Figure 1(a), online VLN requires an agent to perform a sequence of actions within a single test sample and to execute various samples (instructions) in an online manner. On one hand, while conducting TTA at every (or a few) action steps enables rapid agent adaptation to dynamic environments, frequent model updates may introduce significant model alterations, potentially causing cumulative errors and catastrophic forgetting (Wang et al., 2022a; Niu et al., 2022; Song et al., 2023), thus compromising model stability during testing. On the other hand, initializing the same model for stable TTA in each test sample may hinder the model's ability to adaptively learn experience from historical test samples, thereby impeding its potential for achieving superior performance. Figure 1(b) shows that both overly fast or overly slow model updates fail to achieve significant performance improvements.

To tackle the above issues, we propose a Fast-Slow Test-Time Adaptation (FSTTA) method for online VLN tasks. Built upon a unified gradient-parameter decomposition-accumulation framework, our approach consists of a fast update phase and a slow update phase, pursuing a balance between adaptability and stability in online model updating. Specifically, with a test-time training objective, such as entropy minimization (Wang et al., 2021), we can derive gradients at each action step in the fast update phase. However, due to the unsupervised nature of TTA, these gradients inevitably contain noise information. Using these gradients for model update can interfere with the adaptability, especially when the update is frequently invoked. Therefore, we establish a local coordinate system to find a reliable optimization direction by periodically analyzing the gradients generated during the recent multi-step navigation process. After a certain number of fast updates, the model parameters (also called model state) are recorded. To further mitigate the issues of cumulative errors and catastrophic forgetting that may result from excessively frequent model updates, during the slow update phase, we revert the model to its historical state and conduct a decomposition-accumulation analysis on the parameter variation trajectory for a direct model update. Both phases are performed alternately during testing to balance the adaptability and stability of the model. As shown in Figure 1(b), the proposed method achieves significant improvement against other model update strategies.

Our contributions can be summarized as follows: (1) Considering the characteristics of inter-sample online instruction-execution and intra-sample multi-step action-execution, we explore the online VLN task and propose a fast-slow test-time adaptation (FSTTA) method for effective navigation and model updating. (2) Based on a unified decomposition-accumulation framework for both gradients and parameters, our method ensures swift model adaptability to environmental changes in the short-term fast update phase, while preserves stability throughout the long-term slow update phase. (3) Our FSTTA elevates the performance of several leading VLN models on four popular benchmarks. When applied to the notable DUET model (Chen et al., 2022d), it yields a performance boost of over 5% on the representative discrete/continuous benchmarks REVERIE/R2R-CE. Furthermore, our method shows superior results compared to other premier TTA techniques.

2. Related Work

Vision-and-Language Navigation (VLN). Most existing methods facilitate VLN research by developing powerful techniques for model training, including: **(i)** Designing advanced network architectures. Early VLN models utilize LSTMs with various attention mechanisms (Anderson et al., 2018; Fried et al., 2018; Hong et al., 2020) while recent ones (Hao et al., 2020; Hong et al., 2021; Chen et al., 2021; Lin et al., 2022; Chen et al., 2022d; Huo et al., 2023) resort to the more popular transformer-based methods for multi-modal pre-training. Other architectures are also explored such as graph neural networks (Zhu et al., 2021) and parameter-efficient adapter (Qiao et al., 2023b). **(ii)** Adopting various training paradigms such as reinforcement and imitation learning (Nguyen et al., 2019; Tan et al., 2019; Wang et al., 2019; Gao et al., 2023a). Moreover, to estimate the completeness of instruction following and decide when to conduct backtracking, progress monitoring (Ma et al., 2019a; Zhu et al., 2020) and back-tracking (Ke et al., 2019; Ma et al., 2019b) are also employed to promote training process. **(iii)** Performing data augmentation for training a stronger model. In recent years, more and more large-scale benchmarks are established via collecting human annotations (Ku et al., 2020; Zhu et al., 2021; Ramrakhya et al., 2022) or creating new environments (Qi et al., 2020; Chen et al., 2022c). Other approaches explore techniques such as mixup and synthesis (Liu et al., 2021; Kamath et al., 2023), style transfer (Li et al., 2022), or future-view image semantics (Li & Bansal, 2023) for data augmentation. **(iv)** Leveraging additional information for boosting model capacity. Since the goal of VLN is to navigate in photo-realistic environments, there are many kinds of information in the world that can be used such as knowledge (Li et al., 2023), 3D scene geometry (Liu et al., 2023; Wang et al., 2023f), and landmarks (Wang et al., 2022b; Cui et al., 2023).

Although the above methods have made significant progress in training effective models, they overlook the utilization of test data during the online VLN process. In real-world applications, an agent is required to continuously execute user instructions at different times. The ability of an agent to accumulate experience during this process would greatly enhance its practical value. Note that (Lu et al., 2022) first explored test-time adaptation for VLN. However, this approach does not perform model update in an online manner and overlook the balance between adaptability-stability.

Online Test-time Adaptation (TTA). TTA allows models to adapt the test data in an online and unsupervised manner (Liang et al., 2023; Lim et al., 2023; Lee et al., 2023). Existing TTA methods generally rely on batch normalization calibration (Mirza et al., 2022; Zhao et al., 2023; Gong et al., 2022), entropy minimization (Wang et al., 2021; Niu et al., 2023; Tang et al., 2023), auxiliary self-supervised task or data regularization (Sun et al., 2020; Boudiaf et al., 2022; Zhang et al., 2022) to acquire useful information for reducing the domain gap between training and testing data. To stabilize adaptation in continuously changing data distribution, recently, continual test-time adaptation (Wang et al., 2022a; Niu et al., 2022; Song et al., 2023; Döbler et al., 2023; Yuan et al., 2023; Liu et al., 2024), as a more practical setting, has been tentatively explored for addressing the cumulative errors and catastrophic forgetting issues. Until now, test-time adaptation has been preliminarily explored in some sequential data analysis fields such as action recognition (Lin et al., 2023) and video classification (Yi et al., 2023). However, they overlook the joint inter- and intra-sample structure in TTA of sequential data.

Gradient-based Methods. Gradients are central to modern SGD-based deep learning algorithms. To date, gradient analysis research has predominantly focused on domain generalization (DG) (Mansilla et al., 2021; Lew et al., 2023; Wang et al., 2023b; Rame et al., 2022; Wang et al., 2023e; Tian et al., 2023), due to the negative impact of conflicting gradients from multiple domains on model optimization. Pioneering works (Du et al., 2018; Yu et al., 2020; Mansilla et al., 2021) perform gradient surgery at the backpropagation phase via various strategies such as normal plane projection (Yu et al., 2020) and consensus learning (Mansilla et al., 2021). Other approaches resort to gradient agreement regularization for refining the optimization direction by leveraging sharpness (Wang et al., 2023b) or similarity (Shi et al., 2021; Rame et al., 2022) measurements. Different from the above strategies that only consider a single-phase gradient surgery in DG, we jointly analyze the gradient-parameter states for a two-phase (fast-slow) TTA in the VLN task.

3. Our Approach

Problem Setup and VLN Base Model. Given a natural language instruction $\boldsymbol{I}$, the VLN task requires an agent to find the target viewpoint through the environment by executing a series of actions. During the navigation process, an undirected exploration graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$ is progressively constructed, where $\mathcal{V}_t$ denotes navigable nodes, $\mathcal{E}_t$ indicates the connectivity edges, t is the current timestep. Note that a 'STOP' node is added to this graph to indicate a stop action, and we connect it with all other nodes. At this moment, the agent receives a panoramic view that contains 36 single images. The panorama is represented by the image features $\mathcal{R}_t$ and their object features $\mathcal{O}_t$, where these features can be extracted by pre-trained vision transformers (ViT) (Dosovitskiy et al., 2020; Chen et al., 2022d; Li et al., 2023). To accomplish the instruction, the agent needs to predict probabilities for the currently navigable nodes and select the most possible one as the next movement action. The probabilities can be predicted as:

$$\boldsymbol{s}_t = \phi\left(\boldsymbol{I}, \mathcal{R}_t, \mathcal{O}_t, \mathcal{H}_t; \boldsymbol{\Theta}\right), \quad \boldsymbol{s}_t \in \mathbb{R}^{|\mathcal{V}_t|} \tag{1}$$

where $\mathcal{H}_t$ indicates the history information that encodes the observed visual features and performed actions (Chen et al., 2022d; Qiao et al., 2023b). $\phi(\cdot)$ is the VLN base model such as the dual-scale graph transformer (Chen et al., 2022d;c), $\boldsymbol{\Theta}$ is the learnable model parameters.

Framework Overview. In this paper, we devote to adjusting the VLN base model during testing process within an online and unsupervised manner. Our FSTTA framework is illustrated in Figure 2. For each sample, at timestep t, we employ the commonly adopted entropy minimization objective (Wang et al., 2021; Niu et al., 2023) for test-time adaption, which aims to reduce the entropy of the probabilities over the current navigable nodes:

$$\mathcal{L}(\boldsymbol{s}_t; \boldsymbol{\Theta}) = -\sum\nolimits_i \boldsymbol{s}_{t,i} \log(\boldsymbol{s}_{t,i}). \tag{2}$$

During the optimization of the above objective, gradients are back-propagated for updating the model's parameters. However, updating the whole base model is computationally infeasible. As a result, we only consider a small portion of the model parameters for gradient calculation. Since affine parameters in normalization layers capturing data distribution information, numerous TTA methods opt to update these parameters for adaption (Wang et al., 2021; Niu et al., 2023; Liang et al., 2023). In this paper, we employ the model's a few final layer-norm operations for TTA and maintain other parameters frozen. For brevity, we still use the symbol $\boldsymbol{\Theta}$ to represent these parameters to be updated in the following sections, $\boldsymbol{\Theta} \in \mathbb{R}^D$. Targeting at fully leveraging the gradient and parameter information, under a unified decomposition-accumulation analysis framework, we propose an effective two-phase adaptation for fast and slow model updates.

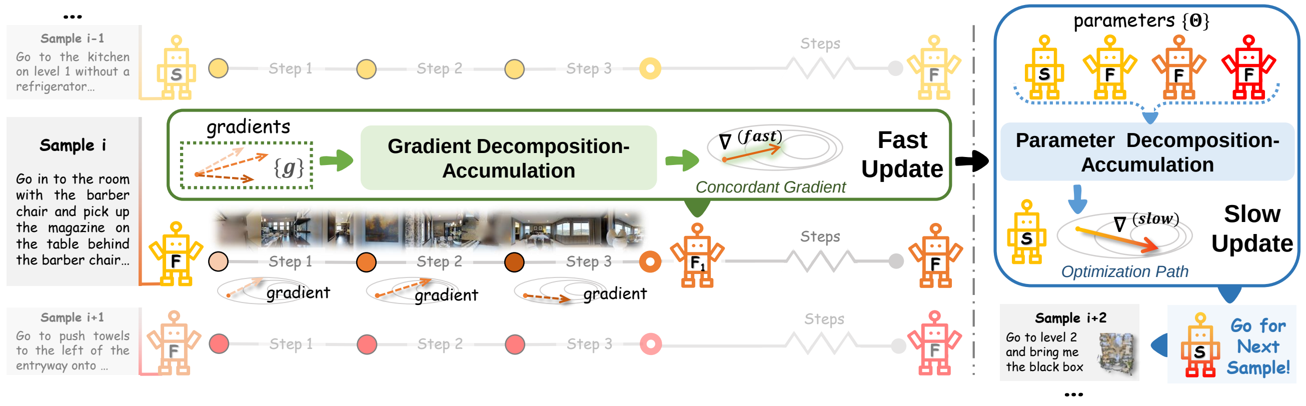

Sample i-1
Go to the kitchen on level 1 without a refrigerator...
Sample i
Go in to the room with the barber chair and pick up the magazine on the table behind the barber chair...
Sample i+1
Go to push towels to the left of the entryway onto ...
Step 1
Step 2
Step 3
Steps
gradients
Gradient Decomposition-Accumulation
Concordant Gradient
Fast Update
gradient
parameters {Θ}
Parameter Decomposition-Accumulation
Slow Update
Optimization Path
Sample i+2
Go to level 2 and bring me the black box
Go for Next Sample!

Figure 2. Overall framework of the proposed Fast-Slow Test-Time Adaptation (FSTTA) for online VLN. In the fast update phase, taking 'Sample i' as an example, the model periodically analyzes the gradients ($\{\boldsymbol{g}\}$) generated during the recent multi-step navigation and performs a gradient decomposition-accumulation analysis to pinpoint a concordant direction for model update. After a certain number of fast updates, historical model parameters ($\{\boldsymbol{\Theta}\}$) are recorded. In the slow update phase, we revert the model to its historical state and conduct a parameter decomposition-accumulation analysis to learn an optimization path for direct parameter modulation. Note that 'F', 'S' in the robots means the model parameters after fast and slow updates. 'F_1' indicates the first fast update within a test sample.

3.1. Fast Update via Gradient Analysis

At timestep t in one navigation process, the agent is required to select an action (navigable node) by using the predicted score $\boldsymbol{s}_t$. With $\boldsymbol{s}_t$, we can calculate the TTA loss (Eq. (2)) and then derive the gradient of the model parameters $\boldsymbol{\Theta}$ as: $\boldsymbol{g}_t = \nabla\mathcal{L}(\boldsymbol{s}_t;\boldsymbol{\Theta})$, $\boldsymbol{g}_t \in \mathbb{R}^D$. Traditional TTA methods conduct adaptation independently at each time step, which can exacerbate the issue of cumulative errors (Niu et al., 2022; Song et al., 2023), particularly in the VLN process that requires frequent action execution. Therefore, we propose to conduct a gradient decomposition-accumulation analysis, wherein we periodically analyze the gradients generated during the recent multi-step navigation process and identify a concordant direction for an iteration of model update.

Gradient Decomposition-Accumulation. During navigation, as shown in Figure 2, we perform model update every M action steps. For the j-th update, gradients from previous M steps are collected as $\boldsymbol{G}_j = \{\widetilde{\boldsymbol{g}}_{j,m}\}_{m=1}^M$, where $\boldsymbol{G}_j \in \mathbb{R}^{M\times D}$, $\widetilde{\boldsymbol{g}}_{j,m}$ indicates the t-th gradient $\boldsymbol{g}_t$ when $t = M(j-1)+m$. Note that these gradients determine the learning direction of our VLN model, and a simple strategy to compute this direction is to take their average $\bar{\boldsymbol{g}}_j = 1/M\sum_m \widetilde{\boldsymbol{g}}_{j,m}$; however, this inevitably introduce step-specific noise. To avoid the issue, we aim to find a concordant direction among these gradients. We first establish a local coordinate system with D orthogonal and unit axes (bases) $\boldsymbol{U}_j = \{\boldsymbol{u}_{j,d}{}^\top\}_{d=1}^D \in \mathbb{R}^{D\times D}$ for gradient decomposition, where each gradient can be approximately linearly represented by these bases. Intuitively, the axes along which the gradients exhibit the higher variance after projection represent the directions of gradients with the lower consistency. These directions have the potential to introduce interference in determining a model update direction. Therefore, it is advisable to reduce the projection of gradients in these directions. To solve the bases $\boldsymbol{U}_j$, we can utilize singular value decomposition (SVD) as follows:

$$\lambda_{j,d}, \boldsymbol{u}_{j,d} = \mathbf{SVD}_d\left(\frac{1}{M-1}\hat{\boldsymbol{G}}_j^\top\hat{\boldsymbol{G}}_j\right), \tag{3}$$

where $\hat{\boldsymbol{G}}_j$ is the centered gradient matrix by removing the mean from $\boldsymbol{G}_j$. The m-th row in $\hat{\boldsymbol{G}}_j$ reflects the deviation between $\widetilde{\boldsymbol{g}}_{j,m}$ and the average gradient $\bar{\boldsymbol{g}}_j$. $\lambda_{j,d}$, $\boldsymbol{u}_{j,d}$ denote the d-th largest eigenvalue and the corresponding eigenvector. Motivated by the principle component analysis (Shlens, 2014), it is obvious that a larger $\lambda_{j,d}$ corresponds to a higher variance of the gradient projection length $\boldsymbol{G}_j\boldsymbol{u}_{j,d}$ and vice versa. Hence, we can derive a concordant gradient by adaptively aggregating the gradients' components on all the axes by considering different eigenvalue (importance):

$$\nabla_j^{(fast)} = \sum\nolimits_{d=1}^{D} \Phi_d(\lambda_{j,d})\cdot <\bar{\boldsymbol{g}}_j, \boldsymbol{u}_{j,d}> \boldsymbol{u}_{j,d}, \tag{4}$$

where the last term denotes the projected component of the averaged gradient $\bar{\boldsymbol{g}}_j$ on to the d-th axis. $\Phi_d(\cdot)$ is referred as the adaptive coefficient for accumulating all the components, which is simply defined as $\Phi_d(\lambda_{j,d}) = 1/\lambda_{j,d}$, reflecting the importance of various axes. Notably, when removing the coefficient, $\nabla_j^{(fast)}$ is degenerated into $\bar{\boldsymbol{g}}_j$, which is used in regular gradient descent approaches.

Based on Eq. (4), a concordant optimization direction is established by enhancing the components that are convergent among $\{\widetilde{\boldsymbol{g}}_{j,m}\}_{m=1}^{M}$ and suppressing those divergent ones. However, the introduction of $\Phi_d(\cdot)$ makes the length of $\nabla_j^{(fast)}$ uncontrollable. Therefore, we calibrate its length to $\|\bar{\boldsymbol{g}}_j\|_2$, which encodes the gradient length from the last three time steps, for a more reasonable model update:

$$\nabla_j^{(fast)} \leftarrow (\nabla_j^{(fast)} \|\bar{\boldsymbol{g}}_j\|_2)/\|\nabla_j^{(fast)}\|_2 \tag{5}$$

With $\nabla^{(fast)}$, we can perform fast model update by setting a learning rate $\gamma^{(fast)}$. Although traditional methods employ a fixed learning rate, such a setting might hinder model convergence, *i.e.*, small learning rates slow down convergence while aggressive learning rates prohibit convergence (Barzilai & Borwein, 1988). Since fast updates are frequently invoked during navigation, relying on a fixed learning rate is sub-optimal. Therefore, we propose to dynamically adjust learning rate throughout the fast update phase.

Dynamic Learning Rate Scaling. Different from varying the learning rate through optimizer or scheduler, we argue for a scaling method that leverages gradient agreement information in historical steps to dynamically adjust the speed of model update. Current gradient alignment strategies typically impose direct constraints on the gradients (Shi et al., 2021; Rame et al., 2022), which are not suitable for our framework as they undermine the gradient decomposition-accumulation process. Given that the second-order information (variance) has been demonstrated to be more effective than the first-order information (mean) in gradient agreement learning (Rame et al., 2022), we directly utilize the trace of the gradient covariance matrix, Tr $\left(1/(M-1)\hat{\boldsymbol{G}}_j^{\top}\hat{\boldsymbol{G}}_j\right)$, for scaling. Note that the trace is equal to the sum of eigenvalues $\sigma_j = \sum_d \lambda_{j,d}$. Here, when σ_j deviates significantly from the historical variance, we assign a smaller learning rate, and vice versa:

$$\gamma_j^{(fast)} = \text{Trunc}\left(1 + \tau - |\sigma_j - \bar{\sigma}|\right) \cdot \hat{\gamma}^{(fast)}, \tag{6}$$

where Trunc$(\cdot)$ is the truncation function that truncates the input to the interval $[a, b]$. τ is a threshold and $\hat{\gamma}^{(fast)}$ is the base learning rate. The historical variance $\bar{\sigma}$ is updated as $\bar{\sigma} \leftarrow \rho\bar{\sigma} + (1-\rho)\sigma_j$ and maintained for all samples throughout the test stage, ρ is the update momentum.

Model Update. With the above gradient and learning rate, we can perform a single iteration as the j-th fast update:

$$\boldsymbol{\Theta}_j = \boldsymbol{\Theta}_{j-1} - \gamma_j^{(fast)} \cdot \nabla_j^{(fast)}, \tag{7}$$

where the subscript of $\boldsymbol{\Theta}$ indicates the index of model update in the current test sample.

3.2. Slow Update via Parameter Analysis

In the fast update phase, although we obtain concordant optimization directions, the frequent parameter updates may still dramatically change the VLN model. To maintain the stability of the VLN model during online long-term usage, we periodically revert the model to its historical states, and conduct a decomposition-accumulation analysis on the parameter variation trajectory for direct parameter modulation. The slow update phase shares the core formulation with the fast phase, but shifts the focus from gradients to the model parameters themselves.

Parameter Decomposition-Accumulation. Following the completion of the fast update phase on the o-th test sample, the model state (parameters) is recorded as $\boldsymbol{\Theta}_{o,J_o}$, where J_o denotes the final fast update step on this sample, and the subscript o has been omitted in the previous section. We then treat these historical states as a parameter variation trajectory to facilitate stable model updates. As shown in the right part of Figure 2, the slow model update is invoked every N samples. For the l-th update, historical model states are collected as $\boldsymbol{M}_l = \{\widetilde{\boldsymbol{\Theta}}_{l,n}\}_{n=0}^{N}$, where $\boldsymbol{M}_l \in \mathbb{R}^{(N+1)\times D}$, $\widetilde{\boldsymbol{\Theta}}_{l,n}$ indicates the o-th model state $\boldsymbol{\Theta}_{o,J_o}$ when $o = N(l-1) + n$ and $n \neq 0$. $\widetilde{\boldsymbol{\Theta}}_{l,0}$ indicates the model state produced by the previous slow update, and we use it interchangeably with $\boldsymbol{\Theta}^{(l-1)}$ in the following. Note that in the slow update phase, we additionally incorporate $\boldsymbol{\Theta}^{(l-1)}$ from the previous update for analysis since it serves as a starting reference point for direct parameter modulation.

Similar to the fast update phase, the centered parameter matrix $\hat{\boldsymbol{M}}_l$ can be constructed, where the n-th row vector in it reflects the deviation between $\widetilde{\boldsymbol{\Theta}}_{l,n}$ and the averaged historical parameter $\bar{\boldsymbol{\Theta}}_l = 1/(N+1)\sum_n \widetilde{\boldsymbol{\Theta}}_{l,n}$. With $\hat{\boldsymbol{M}}_l$, we can obtain the following eigenvalues and eigenvectors: $\epsilon_{l,d}, \boldsymbol{z}_{l,d} = \textbf{SVD}_d(1/N \cdot \hat{\boldsymbol{M}}_l^{\top}\hat{\boldsymbol{M}}_l)$, where a larger $\epsilon_{l,d}$ corresponds to a higher variance of the parameter projection length $\boldsymbol{M}_l\boldsymbol{z}_{l,d}$ and vice versa. $\boldsymbol{Z}_l = \{\boldsymbol{z}_{l,d}{}^{\top}\}_{d=1}^{D}$ depicts the local coordinate system where each axis depicts the direction of parameter variation. Intuitively, the principal axes (with larger eigenvalues) delineate the primary directions of historical parameter variation, while minor axes (with smaller eigenvalues) often encompass noise (Wang et al., 2023e). To find a reliable optimization path to traverse the trajectory of primary parameter changes, we pay attention on the axis with the large variance. Since there is no silver bullet to learning an optimization direction with only parameters, a reference direction can significantly aid in guiding the model towards a local optimal. Here, we leverage the parameter variations to calculate the reference direction:

$$\boldsymbol{h}_l = \frac{1}{\sum_{i=0}^{N-1} q^i} \sum\nolimits_{n=1}^{N} q^{N-n} \cdot (\widetilde{\boldsymbol{\Theta}}_{l,0} - \widetilde{\boldsymbol{\Theta}}_{l,n}), \tag{8}$$

where the hyper-parameter $q \in (0, 1)$, which assigns larger weight to the more recent parameter deviations as they encapsulate richer sample information. Then, we calculate the

Table 1. Experimental results for different TTA strategies on REVERIE dataset.

Methods	**REVERIE** Val Seen				**REVERIE** Val Unseen				**REVERIE** Test Unseen				Time(ms)[*]
	OSR	SR	SPL	RGSPL	OSR	SR	SPL	RGSPL	OSR	SR	SPL	RGSPL	
DUET (Chen et al., 2022d)	73.86	71.75	63.94	51.14	51.07	46.98	33.73	23.03	56.91	52.51	36.06	22.06	104.84
+ EATA (Niu et al., 2022)	74.05	72.17	63.41	49.94	52.09	47.40	33.46	22.65	57.21	52.91	35.19	21.65	133.12
+ CoTTA (Wang et al., 2022a)	73.79	71.05	62.91	49.36	52.46	47.56	31.43	21.83	56.14	52.52	34.66	21.06	3.89×10^3
+ NOTE (Gong et al., 2022)	74.76	72.43	64.28	51.97	52.85	48.28	33.98	22.98	57.66	**53.41**	36.18	22.09	137.89
+ SAR (Niu et al., 2023)	74.84	71.75	64.43	51.70	53.26	48.00	33.92	23.09	57.11	53.04	36.07	22.27	145.53
+ ViDA (Liu et al., 2024)	73.99	72.49	63.49	50.89	52.53	48.14	32.45	21.92	56.78	52.74	35.10	21.77	3.97×10^3
+ Tent (Wang et al., 2021)	70.07	70.73	61.67	49.31	49.43	46.87	31.90	20.15	54.58	50.56	33.37	20.32	126.91
+ Tent-INT-2	71.82	70.36	61.53	49.82	51.22	48.46	33.67	21.30	54.25	50.36	33.89	21.09	124.02
+ Tent-INT-3	74.54	72.58	64.44	51.05	52.28	48.60	34.65	23.12	56.66	52.92	35.84	21.89	119.34
+ Tent-INT-4	73.84	72.51	64.03	50.97	51.40	48.91	35.06	22.99	56.74	53.14	36.29	22.14	117.26
+ Tent-Stable	73.72	71.89	64.06	50.41	51.43	47.55	33.99	23.32	57.12	52.61	36.17	22.16	129.22
+ FSTTA	**75.59**	**75.48**	**65.84**	**52.23**	**56.26**	**54.15**	**36.41**	**23.56**	**58.44**	53.40	**36.43**	**22.40**	135.61

[*] Note that the last column displays the average execution time of the agent for a single instruction, calculated on the Validation Unseen set of REVERIE.

optimization path (gradient) in the slow update phase as:

$$\nabla_l^{(slow)} = \sum\nolimits_d \Psi_d(\boldsymbol{\epsilon}_l, \boldsymbol{h}_l) \cdot \text{sign}\left(< \boldsymbol{h}_l, \boldsymbol{z}_{l,d} >\right) \boldsymbol{z}_{l,d}, \quad (9)$$

where the use of sign function $\text{sign}(\cdot)$ is to force the axes to be positively related to the reference direction $\boldsymbol{h}_l$. Notably, different from Eq. (4) that uses the projected components on each axis for estimating an optimization direction, here we only utilize the axes themselves for deriving $\nabla_l^{(slow)}$. The reason is that these axes depict the parameter variation direction, which can be directly used for estimating gradients. $\Psi_d(\cdot)$ is referred as the adaptive coefficient for accumulating all the axes (optimization directions), defined as:

$$\Psi_d(\boldsymbol{\epsilon}_l, \boldsymbol{h}_l) = \frac{\epsilon_{l,d} \cdot \|\boldsymbol{h}_l\|_2}{\|\boldsymbol{\epsilon}_l\|_2}, \quad (10)$$

where the L2-normalization is performed on eigenvalues to convey different relative importance of axes. Besides, the norm of the reference direction is utilized to automatically tuning the magnitude of the analyzed gradient. In contrast to $\Phi_d(\cdot)$ in the fast update phase, $\Psi_d(\cdot)$ highlights those axes with high variation due to the different characteristics of gradients and parameters in model optimization.

Model Update. With $\nabla_l^{(slow)}$, we can perform the l-th slow model update as follows:

$$\boldsymbol{\Theta}^{(l)} = \boldsymbol{\Theta}^{(l-1)} - \gamma^{(slow)} \cdot \nabla_l^{(slow)}, \quad (11)$$

where $\gamma^{(slow)}$ is learning rate. Since the slow update phase is designed for stable model learning and is not frequently invoked, we employ a fixed learning rate here instead of conducting the dynamic learning rate scaling as done in the fast phase. The updated parameter $\boldsymbol{\Theta}^{(l)}$ will be utilized for the subsequently coming test samples in conjunction with new fast update phases applied to them.

4. Experimental Results

Datasets. We use the popular and standard VLN benchmark REVERIE (Qi et al., 2020) to investigate test-time adaptation within the realm of online VLN. REVERIE contains 10,567 panoramic images and 21,702 high-level instructions, focusing on grounding remote target object within 90 buildings. In addition, we also adopt other three benchmarks for evaluating the effectiveness of our proposed FSTTA. Among them, R2R (Anderson et al., 2018) provides step-by-step instructions for navigation in photo-realistic environments, which includes 10,800 panoramic views and 7,189 trajectories. SOON (Zhu et al., 2021) also requires the agent to find the target object with a more detailed description of the goal. It has 3,848 sets of instruction and more than 30K long distance trajectories. Note that the performance comparisons are based on their challenge report, specifically within the provided val unseen and test unseen splits. R2R-CE (Krantz et al., 2020) is a variant of R2R in continuous environments, where an agent is able to move freely and engage with obstacles. The dataset consists of 16,000 instruction-trajectory pairs, with non-transferrable paths excluded.

Evaluation Metrics. We follow previous approaches (Qi et al., 2020; Chen et al., 2022d;c; Li et al., 2022; Wang et al., 2023f) and employ the most commonly used metrics for evaluating VLN agents, i.e., TL (Trajectory Length): the agent's average path length in meters; NE (Navigation Error): average distance in meters between the agent's final location and the target one; SR (Success Rate): the proportion of successfully executed instructions with the NE less than 3 meters; SPL (Success weighted by Path Length): SR penalized by Path Length, which is calculated as $\frac{1}{E}\sum_{i=1}^{E} S_i \frac{l_i}{max(p_i, l_i)}$, where E is the number of tasks, S_i denotes the success as a binary value, l_i and p_i denote the shortest path and actual path length for the i^{th} task; OSR (Oracle Success Rate): SR given the oracle stop policy; RGS (Remote Grounding Success rate): proportion of successfully executed instructions where the output bounding

Table 2. Ablation study on REVERIE dataset.

Module			REVERIE Val Unseen					
Fast	DLR	Slow	TL ↓	OSR	SR	SPL	RGS	RGSPL
-	-	-	**22.11**	51.07	46.98	33.73	32.15	23.03
Tent	-	-	22.52	52.28	48.60	34.65	32.66	23.12
✓	-	-	22.65	53.50	49.74	34.91	33.70	23.36
✓	✓	-	22.43	54.01	49.82	35.34	**34.32**	23.29
✓	✓	✓	22.14	**56.26**	**54.15**	**36.41**	34.27	**23.56**

Table 3. Results on Validation Seen set of REVERIE.

FSTTA		REVERIE Val Seen					
Unseen	Seen	TL ↓	OSR	SR	SPL	RGS	RGSPL
×	×	13.86	73.86	71.15	63.94	57.41	51.14
× ⟶	✓	15.13	**75.59**	**75.48**	**65.84**	58.62	**52.23**
✓ ⟶	×	**13.40**	73.16	71.78	64.18	57.05	51.18
✓ ⟶	✓	15.11	75.58	74.12	65.53	**59.20**	52.18

Table 4. Results on Validation Unseen & Seen sets of REVERIE.

Methods	REVERIE Val Unseen & Seen					
	TL ↓	OSR	SR	SPL	RGS	RGSPL
DUET	**19.18**	61.53	57.49	45.66	41.56	34.38
+ Tent	20.23	57.33	54.86	41.90	38.09	32.46
+ EATA	20.29	62.77	57.31	44.59	41.54	34.16
+ SAR	20.52	**63.59**	57.80	44.72	42.45	34.88
+ FSTTA	20.48	63.36	**60.23**	**47.96**	**43.58**	**35.65**

box has an IoU (intersection over union) ≥ 0.5 with the ground truth; and RGSPL (RGS weighted by Path Length): RGS penalized by Path Length. Among them, SR and SPL are the most common metrics for evaluation. Note that only the optimal values of experimental results are highlighted in bold across all tables. Moreover, from Tables 5 to 8, different font colors are employed to indicate whether our method exceeds the performance of the corresponding baseline methods, *i.e.*, using red to denote superior results and blue for inferior ones.

Implementation Details. To better conform to practical applications, we set batch size to 1 during evaluation, where each sample (and each action step) is forward propagated only once. Owing to the lack of an authentic online VLN evaluation setting, we shuffle test samples in each dataset split and sequentially input them into the agent to simulate the online execution and adaptation. Specifically, for VLN models equipped with TTA strategies, we run the experiments with shuffled samples 5 times and report the average results. We adopt DUET (Chen et al., 2022d) and HM3D (Chen et al., 2022c) as the base models. Since HM3D does not provide training code for R2R-CE, we adopt another SOTA methods, WS-MGMap (Chen et al., 2022b) and BEVBert (An et al., 2022), for TTA. Note that for the base models, we report the results obtained from running their official codes. In our FSTTA, we only utilize the last four LN layers of base models for model updating, all the feature dimensions of these layers are 768. We set the intervals for fast and slow updates to $M = 3$ and $N = 4$, the learning rates of the two phases are $\hat{\gamma}^{(fast)} = 6 \times 10^{-4}$ and $\gamma^{(slow)} = 1 \times 10^{-3}$. For the dynamic learning rate scaling, we empirically set the threshold $\tau = 0.7$ in Eq. (6) and the update momentum $\rho = 0.95$ with the truncation interval $[0.9, 1.1]$. And the hyper-parameter q in Eq. (8) is set to 0.1. All experiments are conducted on a RTX 3090 GPU. See § A for more details.

4.1. Comparison with Different TTA Strategies

Currently, various TTA methods have been adeptly integrated for the dynamic model updates, marking significant progress. Although the exploration of TTA within the VLN field remains relatively untapped, the integration of contemporary advanced TTA methodologies into VLN is feasible. Since efficiency is an important evaluation metric for TTA, we provide the average time taken by each method to execute a single instruction for comparison. For the compared methods, SAR and TENT are the popular entropy minimization models, whereas NOTE, CoTTA, EATA and ViDA are state-of-the-art continual TTA methods. The results in Table 1 demonstrate the capability of our proposed FSTTA to blend model performance with testing efficiency. Specifically, on the validation unseen dataset, our method exhibits a discernible enhancement of 6.2% and 2.5% on the SR and SPL metrics compared to the state-of-the-art SAR method, concurrently manifesting a reduction of 7% in testing time. From the results, we observe that directly applying existing TTA methods to the online VLN task does not lead to significant performance improvements. Furthermore, we investigate different frequencies of updates based on TENT as well as the stable update approach. 'INT' denotes the update interval, which means averaging the gradient information over a certain interval and then performing an iteration of model update; these results are consistent with those in Figure 1(b). It can be seen that our method still outperforms these strategies with marginally increased time costs.

4.2. Extensive Analysis of FSTTA

Ablation Studies of the Proposed FSTTA. In this work, we propose a FSTTA method for online VLN, which consists of both fast and slow model update phases. To validate their effectiveness, we progressively integrate the two phases into the baseline DUET model. In addition, we design a baseline variant, which equips DUET with the vanilla TTA objective (TENT (Wang et al., 2021)) and simply utilize the averaged gradient in an interval (with the same M) for fast model updates. Empirical findings from Table 2 illuminate that the integration of fast and slow phases progressively

Table 5. Experimental results on REVERIE dataset.

Methods	Val Seen				Val Unseen				Test Unseen			
	OSR	SR	SPL	RGSPL	OSR	SR	SPL	RGSPL	OSR	SR	SPL	RGSPL
Seq2Seq (Anderson et al., 2018)	35.70	29.59	24.01	14.96	8.07	4.20	2.84	2.16	6.88	3.99	3.09	1.58
RCM (Wang et al., 2019)	29.44	23.33	21.82	15.36	14.23	9.29	6.97	3.89	11.68	7.84	6.67	3.14
FAST (Qi et al., 2020)	55.17	50.53	45.50	29.66	28.20	14.40	7.19	4.67	30.63	19.88	11.61	6.08
SIA (Lin et al., 2021)	65.85	61.91	57.08	42.65	44.67	31.53	16.28	11.56	44.56	30.80	14.85	9.20
RecBERT (Hong et al., 2021)	53.90	51.79	47.96	35.61	35.02	30.67	24.90	15.27	32.91	29.61	23.99	13.51
Airbert (Guhur et al., 2021)	48.98	47.01	42.34	30.01	34.51	27.89	21.88	14.18	34.20	30.28	23.61	13.28
HAMT (Chen et al., 2021)	47.65	43.29	40.19	25.18	36.84	32.95	30.20	17.28	33.41	30.40	26.67	13.08
HOP (Qiao et al., 2022)	53.76	54.88	47.19	33.85	36.24	31.78	26.11	15.73	33.06	30.17	24.34	14.34
BEVBert (An et al., 2022)	**76.18**	73.72	65.32	51.73	56.40	51.78	36.37	24.44	57.26	52.81	36.41	22.09
LANA (Wang et al., 2023d)	74.28	71.94	62.77	50.34	52.97	48.31	33.86	22.77	57.20	51.72	36.45	22.85
GridMM (Wang et al., 2023f)	-	-	-	-	57.48	51.37	36.47	24.56	59.55	55.13	36.60	**23.45**
DUET (Chen et al., 2022d)	73.86	71.75	63.94	51.14	51.07	46.98	33.73	23.03	56.91	52.51	36.06	22.06
DUET-FSTTA	75.59	**75.48**	**65.84**	**52.23**	56.26	54.15	36.41	23.56	58.44	53.40	36.43	22.40
HM3D (Chen et al., 2022c)	66.76	65.00	55.70	41.66	62.11	55.89	40.85	**26.76**	59.81	53.13	38.24	22.68
HM3D-FSTTA	69.41	67.79	58.28	41.60	**63.74**	**57.02**	**41.41**	26.55	**63.68**	**56.44**	**39.58**	23.04

Table 6. Experimental results on R2R dataset.

Methods	Val Seen				Val Unseen			
	TL ↓	NE ↓	SR	SPL	TL ↓	NE ↓	SR	SPL
Seq2Seq (Anderson et al., 2018)	11.33	6.01	39	-	**8.39**	7.81	22	-
RCM (Wang et al., 2019)	10.65	3.53	67	-	11.46	6.09	43	-
EnvDrop (Tan et al., 2019)	11.00	3.99	62	59	10.70	5.22	52	48
PREVALENT (Hao et al., 2020)	**10.32**	3.67	69	65	10.19	4.71	58	53
RecBERT (Hong et al., 2021)	11.13	2.90	72	68	12.01	3.93	63	57
HAMT (Chen et al., 2021)	11.15	2.51	76	72	11.46	3.65	66	61
HOP (Qiao et al., 2022)	11.26	2.72	75	70	12.27	3.80	64	57
DAVIS (Lu et al., 2022)	12.45	3.16	80	**76**	12.65	3.16	67	61
BEVBert (An et al., 2022)	13.56	**2.17**	**81**	74	14.55	2.81	**75**	**64**
DUET (Chen et al., 2022d)	12.33	2.28	79	73	13.94	3.31	72	60
DUET-FSTTA	13.39	2.25	79	73	14.64	3.03	**75**	62
HM3D (Chen et al., 2022c)	13.30	2.70	77	71	14.29	2.83	74	62
HM3D-FSTTA	13.52	2.57	78	71	14.86	**2.71**	**75**	63

bolsters the base model by 2.8% and 4.3% on the SR metric. Moreover, the dynamic learning rate scaling module (DLR) also contributes to enhancing the model's performance.

Will our method experience catastrophic forgetting? For an online VLN agent endowed with the TTA capability, it faces the issue of catastrophic forgetting of historical environments and instructions upon continually executing new instructions in new environments. To assess whether our method harbors this issue, we re-evaluate our methods on the REVERIE *validation seen* set. Compared with the base model, as shown in Table 3, we find that: (1) Obviously, directly applying FSTTA with the base model on seen data can noticeably enhance performance. (2) After performing FSTTA on the unseen set, the obtained model, when tested directly on the seen dataset without TTA, achieves performance comparable to the base model, confirming that our method does not suffer from catastrophic forgetting. (3) Applying the updated model from unseen set to the seen set with TTA yields comparable results against the seen-only-TTA version. This indicates that our method is effective in environment adaption and experience accumulation.

Generalization Testing in More Practical Environments. In practical applications, agents might encounter both previously seen and unseen scenarios. In preceding experiments, we exclusively test on the validation seen and unseen sets separately. To verify the generalizability, we combine the seen and unseen sets into a unified set for online VLN. Table 4 shows that FSTTA outperforms other TTA methods in effectively managing a variety of testing scenarios.

4.3. Comparison with State-of-the-art VLN Models

REVERIE. Table 5 presents a comparison on REVERIE dataset. Compared with the base models which do not perform test-time adaptation, the proposed method shows favorable performance improvement across most evaluation metrics across the three dataset splits. Specifically, on the validation unseen split, our model exhibits notable advantages over DUET, with improvements of 7.1% on SR, and 2.7% on SPL. These results unequivocally affirm the effectiveness of our fast-slow test time adaptation model, showing the promising potential of TTA in the VLN field.

R2R. Table 6 shows the comparison results on R2R dataset. Our approach outperforms the base models in most metrics (*e.g.*, 72% → 75% for DUET on SR, 62% → 63% for HM3D on SPL). Notably, from the results of the above two datasets, our method, while enhancing the success rate of VLN, causes a slight increase in the path length (TL). We speculate that a possible reason is that performing TTA online may increase the likelihood of the agent deviating from its original action execution pattern, leading to more exploration or backtracking. This situation is also confirmed in the analysis of various TTA strategies in Table 1.

SOON. The proposed FSTTA establishes new state-of-the-art results across most metrics on this dataset. For instance, as shown in Table 7, on the validation unseen split, our

Table 7. Experimental results on SOON dataset.

Methods	Val Unseen				Test Unseen			
	OSR	SR	SPL	RGSPL	OSR	SR	SPL	RGSPL
GBE (Zhu et al., 2021)	28.54	19.52	13.34	1.16	21.45	12.90	9.23	0.45
GridMM (Wang et al., 2023f)	53.39	37.46	24.81	3.91	48.02	36.27	21.25	4.15
KERM (Li et al., 2023)	51.62	38.05	23.16	4.04	-	-	-	-
AZHP (Gao et al., 2023a)	**56.19**	40.71	26.58	**5.53**	-	-	-	-
DUET (Chen et al., 2022d)	50.91	36.28	22.58	3.75	43.00	33.44	21.42	4.17
DUET-FSTTA	52.57	36.53	23.82	3.75	43.44	35.34	23.23	4.52
HM3D (Chen et al., 2022c)	53.22	41.00	30.69	4.06	47.26	40.26	28.09	5.15
HM3D-FSTTA	54.19	**42.44**	**31.03**	4.93	**48.52**	**42.02**	**28.95**	**5.20**

Table 8. Experimental results on R2R-CE dataset.

Methods	Val Unseen				Test Unseen			
	NE ↓	OSR	SR	SPL	NE ↓	OSR	SR	SPL
Seq2Seq (Krantz et al., 2020)	7.37	40	32	30	7.91	36	28	25
Sim2Sim (Krantz & Lee, 2022)	6.07	52	43	36	6.17	52	44	37
CWP-BERT (Hong et al., 2022)	5.74	53	44	39	5.89	51	42	36
DREAMW (Wang et al., 2023a)	5.53	49	59	44	5.48	49	57	44
GridMM (Wang et al., 2023f)	5.11	61	49	41	5.64	56	46	39
ETPNav (An et al., 2023)	4.71	65	57	49	5.12	63	55	48
WS-MGMap (Chen et al., 2022b)	6.28	48	39	34	7.11	45	35	28
WS-MGMap-FSTTA	6.16	49	40	35	7.62	46	37	28
DUET (Chen et al., 2022d)	5.13	55	46	40	5.82	50	42	36
DUET-FSTTA	5.27	58	48	42	5.84	55	46	38
BEVBert (An et al., 2022)	4.57	**67**	59	50	**4.70**	67	59	50
BEVBert-FSTTA	**4.39**	65	**60**	**51**	5.45	**69**	**60**	**50**

model HM3D-FSTTA achieves SR and SPL of 42.44% and 31.03%, respectively, while the state-of-the-art method GridMM are 37.46% and 24.81%. On the test unseen split, our approach improves the performance of DUET by substantial gains (*e.g.*, 21.42% → 23.23% for SPL).

R2R-CE. FSTTA also generalizes well on the continuous environment, *i.e.*, R2R-CE dataset, as shown in Table 8. The results indicate that our approach demonstrates superior or comparable performance against other methods.

Qualitative Analysis. Figure 3 provides a visualization of the agent's instruction execution process, validating that our proposed FSTTA approach can indeed dynamically enhance the VLN performance of the agent during testing.

Other Experiments. Please refer to Appendixes for other experiments, such as detailed results on 4 datasets in § B.1, comprehensive comparison with TTA strategies in § B.2, and parameter analysis in § B.3.

5. Conclusions

This paper explores the feasibility of TTA strategies for online VLN. We propose a fast-slow test-time adaptation method, which performs decomposition-accumulation analysis for both gradients and parameters, achieving a balance between adaptability and stability. The encouraging performance is validated in extensive experiments.

Limitations. Several limitations are noteworthy. Firstly, our approach focuses on adapting normalization layers within the model. While these layers are widely employed in deep learning, there are still a few methods that do not utilize them. One viable approach to address this issue is to introduce additional normalization layers to the corresponding models and retrain them using the training data. In the future, we will also explore how our model can update other types of layers. Secondly, In this paper, we simulate the online VLN setting simply by sequentially inputting data from the test set. In the future, we aim to construct a more realistic agent online learning dataset that aligns with practical application scenarios, to better evaluate TTA methods. Thirdly, compared to the base model, the introduction of TTA inevitably incurs additional computational cost, which is a direction for future improvement. Finally, the frequencies of fast and slow updates are fixed and periodic. Adaptive update invocation strategies is worthy of consideration.

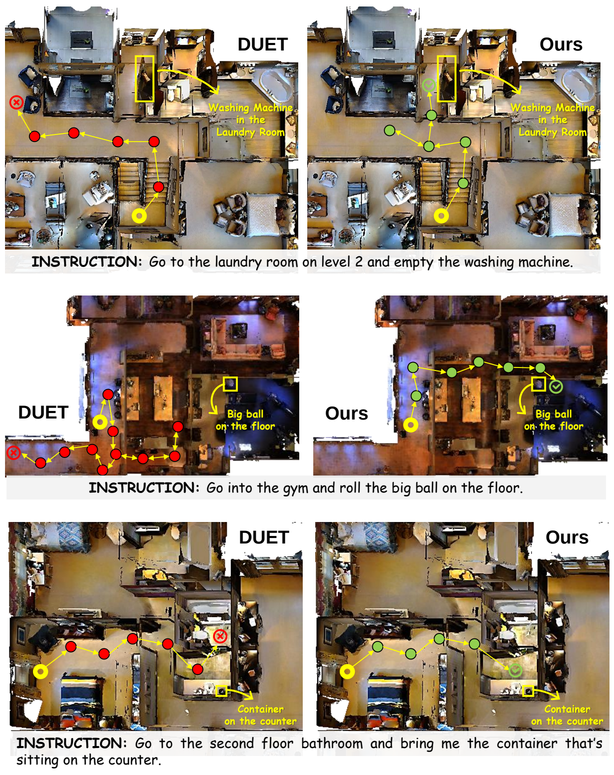

DUET
Ours
Washing Machine in the Laundry Room
INSTRUCTION: Go to the laundry room on level 2 and empty the washing machine.
DUET
Ours
Big ball on the floor
INSTRUCTION: Go into the gym and roll the big ball on the floor.
DUET
Ours
Container on the counter
INSTRUCTION: Go to the second floor bathroom and bring me the container that's sitting on the counter.

Figure 3. Representative visual results on REVERIE validation unseen set. Yellow points denote start locations, while the directed lines with green and red points depict the predicted trajectories with target and incorrect endpoints, respectively. With FSTTA, the basic agent (DUET) demonstrates enhanced exploration capabilities, effectively moving towards the correct direction, and succeeds based on the object context and scene layouts.

Acknowledgements

This work was supported in part by the National Key Research and Development Plan of China under Grant 2023YFC3310700, in part by the National Natural Science Foundation of China under Grants 62036012, 62236008, U21B2044, 62102415, 62072286, and 62106262.

Impact Statement

This paper presents work whose goal is to advance the field of Vision-and-Language Navigation and Online Test-Time Adaptation. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

An, D., Qi, Y., Huang, Y., Wu, Q., Wang, L., and Tan, T. Neighbor-view enhanced model for vision and language navigation. In *ACM MM*, pp. 5101–5109, 2021.

An, D., Qi, Y., Li, Y., Huang, Y., Wang, L., Tan, T., and Shao, J. Bevbert: Topo-metric map pre-training for language-guided navigation. *arXiv preprint arXiv:2212.04385*, 2022.

An, D., Wang, H., Wang, W., Wang, Z., Huang, Y., He, K., and Wang, L. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. *ArXiv*, abs/2304.03047, 2023.

Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I., Gould, S., and Van Den Hengel, A. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, pp. 3674–3683, 2018.

Anderson, P., Shrivastava, A., Parikh, D., Batra, D., and Lee, S. Chasing ghosts: Instruction following as bayesian state tracking. In *NeurIPS*, volume 32, pp. 371–381, 2019.

Barzilai, J. and Borwein, J. M. Two-point step size gradient methods. *Ima Journal of Numerical Analysis*, 8:141–148, 1988.

Boudiaf, M., Mueller, R., Ben Ayed, I., and Bertinetto, L. Parameter-free online test-time adaptation. In *CVPR*, pp. 8344–8353, 2022.

Chen, J., Gao, C., Meng, E., Zhang, Q., and Liu, S. Reinforced structured state-evolution for vision-language navigation. In *CVPR*, pp. 15429–15438, 2022a.

Chen, K., Chen, J., Chuang, J., V'azquez, M., and Savarese, S. Topological planning with transformers for vision-and-language navigation. In *CVPR*, pp. 11271–11281, 2020.

Chen, P., Ji, D., Lin, K.-L. C., Zeng, R., Li, T. H., Tan, M., and Gan, C. Weakly-supervised multi-granularity map learning for vision-and-language navigation. In *NeurIPS*, pp. 38149–38161, 2022b.

Chen, S., Guhur, P.-L., Schmid, C., and Laptev, I. History aware multimodal transformer for vision-and-language navigation. In *NeurIPS*, pp. 5834–5847, 2021.

Chen, S., Guhur, P.-L., Tapaswi, M., Schmid, C., and Laptev, I. Learning from unlabeled 3d environments for vision-and-language navigation. In *ECCV*, pp. 638–655, 2022c.

Chen, S., Guhur, P.-L., Tapaswi, M., Schmid, C., and Laptev, I. Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In *CVPR*, pp. 16537–16547, 2022d.

Cui, Y., Xie, L., Zhang, Y., Zhang, M., Yan, Y., and Yin, E. Grounded entity-landmark adaptive pre-training for vision-and-language navigation. In *ICCV*, pp. 12043–12053, 2023.

Döbler, M., Marsden, R. A., and Yang, B. Robust mean teacher for continual and gradual test-time adaptation. In *CVPR*, pp. 7704–7714, 2023.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020.

Du, Y., Czarnecki, W. M., Jayakumar, S. M., Farajtabar, M., Pascanu, R., and Lakshminarayanan, B. Adapting auxiliary losses using gradient similarity. *arXiv preprint arXiv:1812.02224*, 2018.

Fried, D., Hu, R., Cirik, V., Rohrbach, A., Andreas, J., Morency, L.-P., Berg-Kirkpatrick, T., Saenko, K., Klein, D., and Darrell, T. Speaker-follower models for vision-and-language navigation. In *NeurIPS*, volume 31, 2018.

Gao, C., Chen, J., Liu, S., Wang, L., Zhang, Q., and Wu, Q. Room-and-object aware knowledge reasoning for remote embodied referring expression. In *CVPR*, pp. 3063–3072, 2021a.

Gao, C., Peng, X., Yan, M., Wang, H., Yang, L., Ren, H., Li, H., and Liu, S. Adaptive zone-aware hierarchical planner for vision-language navigation. In *CVPR*, pp. 14911–14920, 2023a.

Gao, J. and Xu, C. Learning video moment retrieval without a single annotated video. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(3):1646–1657, 2021.

Gao, J., Zhang, T., and Xu, C. Learning to model relationships for zero-shot video classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43 (10):3476–3491, 2021b.

Gao, J., Chen, M., and Xu, C. Vectorized evidential learning for weakly-supervised temporal action localization. *IEEE transactions on pattern analysis and machine intelligence*, 45:15949 – 15963, 2023b.

Georgakis, G., Schmeckpeper, K., Wanchoo, K., Dan, S., Miltsakaki, E., Roth, D., and Daniilidis, K. Cross-modal map learning for vision and language navigation. In *CVPR*, pp. 15439–15449, 2022.

Gong, T., Jeong, J., Kim, T., Kim, Y., Shin, J., and Lee, S.-J. Note: Robust continual test-time adaptation against temporal correlation. In *NeurIPS*, pp. 27253–27266, 2022.

Gu, J., Stefani, E., Wu, Q., Thomason, J., and Wang, X. Vision-and-language navigation: A survey of tasks, methods, and future directions. In *ACL*, pp. 7606–7623, 2022.

Guhur, P.-L., Tapaswi, M., Chen, S., Laptev, I., and Schmid, C. Airbert: In-domain pretraining for vision-and-language navigation. In *ICCV*, pp. 1614–1623, 2021.

Hao, W., Li, C., Li, X., Carin, L., and Gao, J. Towards learning a generic agent for vision-and-language navigation via pre-training. In *CVPR*, pp. 13137–13146, 2020.

Hong, Y., Rodriguez-Opazo, C., Qi, Y., Wu, Q., and Gould, S. Language and visual entity relationship graph for agent navigation. In *NeurIPS*, pp. 7685–7696, 2020.

Hong, Y., Wu, Q., Qi, Y., Rodriguez-Opazo, C., and Gould, S. Vln bert: A recurrent vision-and-language bert for navigation. In *CVPR*, pp. 1643–1653, 2021.

Hong, Y., Wang, Z., Wu, Q., and Gould, S. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *CVPR*, pp. 15418–15428, 2022.

Hu, Y., Gao, J., Dong, J., Fan, B., and Liu, H. Exploring rich semantics for open-set action recognition. *IEEE Transactions on Multimedia*, 26:5410 – 5421, 2024.

Huo, J., Sun, Q., Jiang, B., Lin, H., and Fu, Y. Geovln: Learning geometry-enhanced visual representation with slot attention for vision-and-language navigation. In *CVPR*, pp. 23212–23221, 2023.

Irshad, M. Z., Mithun, N. C., Seymour, Z., Chiu, H.-P., Samarasekera, S., and Kumar, R. Semantically-aware spatio-temporal reasoning agent for vision-and-language navigation in continuous environments. In *ICPR*, pp. 4065–4071, 2021.

Kamath, A., Anderson, P., Wang, S., Koh, J. Y., Ku, A., Waters, A., Yang, Y., Baldridge, J., and Parekh, Z. A new path: Scaling vision-and-language navigation with synthetic instructions and imitation learning. In *CVPR*, pp. 10813–10823, 2023.

Ke, L., Li, X., Bisk, Y., Holtzman, A., Gan, Z., Liu, J., Gao, J., Choi, Y., and Srinivasa, S. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *CVPR*, pp. 6741–6749, 2019.

Krantz, J. and Lee, S. Sim-2-sim transfer for vision-and-language navigation in continuous environments. In *ECCV*, pp. 588–603, 2022.

Krantz, J., Wijmans, E., Majumdar, A., Batra, D., and Lee, S. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *ECCV*, pp. 104–120, 2020.

Krantz, J., Gokaslan, A., Batra, D., Lee, S., and Maksymets, O. Waypoint models for instruction-guided navigation in continuous environments. In *ICCV*, pp. 15142–15151, 2021.

Ku, A., Anderson, P., Patel, R., Ie, E., and Baldridge, J. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *EMNLP*, pp. 4392–4412, 2020.

Lee, J., Das, D., Choo, J., and Choi, S. Towards open-set test-time adaptation utilizing the wisdom of crowds in entropy minimization. In *ICCV*, pp. 16380–16389, 2023.

Lew, B., Son, D., and Chang, B. Gradient estimation for unseen domain risk minimization with pre-trained models. In *ICCV*, pp. 4436–4446, 2023.

Li, J. and Bansal, M. Improving vision-and-language navigation by generating future-view image semantics. In *CVPR*, pp. 10803–10812, 2023.

Li, J., Tan, H., and Bansal, M. Envedit: Environment editing for vision-and-language navigation. In *CVPR*, pp. 6741–6749, 2022.

Li, X., Li, C., Xia, Q., Bisk, Y., Celikyilmaz, A., Gao, J., Smith, N. A., and Choi, Y. Robust navigation with language pretraining and stochastic sampling. In *EMNLP*, pp. 1494–1499, 2019.

Li, X., Wang, Z., Yang, J., Wang, Y., and Jiang, S. Kerm: Knowledge enhanced reasoning for vision-and-language navigation. In *CVPR*, pp. 2583–2592, 2023.

Liang, J., He, R., and Tan, T. A comprehensive survey on test-time adaptation under distribution shifts. *arXiv preprint arXiv:2303.15361*, 2023.

Lim, H., Kim, B., Choo, J., and Choi, S. Ttn: A domain-shift aware batch normalization in test-time adaptation. In *ICLR*, 2023.

Lin, C., Jiang, Y., Cai, J., Qu, L., Haffari, G., and Yuan, Z. Multimodal transformer with variable-length memory for vision-and-language navigation. In *ECCV*, pp. 380–397, 2022.

Lin, W., Mirza, M. J., Kozinski, M., Possegger, H., Kuehne, H., and Bischof, H. Video test-time adaptation for action recognition. In *CVPR*, pp. 22952–22961, 2023.

Lin, X., Li, G., and Yu, Y. Scene-intuitive agent for remote embodied visual grounding. In *CVPR*, pp. 7032–7041, 2021.

Liu, C., Zhu, F., Chang, X., Liang, X., and Shen, Y.-D. Vision-language navigation with random environmental mixup. In *ICCV*, pp. 1624–1634, 2021.

Liu, J., Yang, S., Jia, P., Lu, M., Guo, Y., Xue, W., and Zhang, S. Vida: Homeostatic visual domain adapter for continual test time adaptation. In *ICLR*, 2024.

Liu, R., Wang, X., Wang, W., and Yang, Y. Bird's-eye-view scene graph for vision-language navigation. In *ICCV*, pp. 10968–10980, 2023.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *ICLR*, 2017.

Lu, Y., Zhang, H., Nie, P., Feng, W., Xu, W., Wang, X. E., and Wang, W. Y. Anticipating the unseen discrepancy for vision and language navigation. *arXiv preprint arXiv:2209.04725*, 2022.

Ma, C.-Y., Lu, J., Wu, Z., AlRegib, G., Kira, Z., Socher, R., and Xiong, C. Self-monitoring navigation agent via auxiliary progress estimation. In *ICLR*, 2019a.

Ma, C.-Y., Wu, Z., AlRegib, G., Xiong, C., and Kira, Z. The regretful agent: Heuristic-aided navigation through progress estimation. In *CVPR*, pp. 6732–6740, 2019b.

Mansilla, L., Echeveste, R., Milone, D. H., and Ferrante, E. Domain generalization via gradient surgery. In *ICCV*, pp. 6630–6638, 2021.

Mirza, M. J., Micorek, J., Possegger, H., and Bischof, H. The norm must go on: Dynamic unsupervised domain adaptation by normalization. In *CVPR*, pp. 14765–14775, 2022.

Nguyen, K., Dey, D., Brockett, C., and Dolan, B. Vision-based navigation with language-based assistance via imitation learning with indirect intervention. In *CVPR*, pp. 12527–12537, 2019.

Niu, S., Wu, J., Zhang, Y., Chen, Y., Zheng, S., Zhao, P., and Tan, M. Efficient test-time model adaptation without forgetting. In *ICML*, pp. 16888–16905, 2022.

Niu, S., Wu, J., Zhang, Y., Wen, Z., Chen, Y., Zhao, P., and Tan, M. Towards stable test-time adaptation in dynamic wild world. In *ICLR*, 2023.

Qi, Y., Wu, Q., Anderson, P., Wang, X., Wang, W. Y., Shen, C., and Hengel, A. v. d. Reverie: Remote embodied visual referring expression in real indoor environments. In *CVPR*, pp. 9982–9991, 2020.

Qi, Y., Pan, Z., Hong, Y., Yang, M.-H., van den Hengel, A., and Wu, Q. The road to know-where: An and-room informed sequential bert for indoor vision-language navigation. In *ICCV*, pp. 1635–1644, 2021.

Qiao, Y., Qi, Y., Hong, Y., Yu, Z., Wang, P., and Wu, Q. Hop: History-and-order aware pretraining for vision-and-language navigation. In *CVPR*, pp. 15397–15406, 2022.

Qiao, Y., Qi, Y., Yu, Z., Liu, J., and Wu, Q. March in chat: Interactive prompting for remote embodied referring expression. In *ICCV*, pp. 15758–15767, 2023a.

Qiao, Y., Yu, Z., and Wu, Q. Vln-petl: Parameter-efficient transfer learning for vision-and-language navigation. In *ICCV*, pp. 15443–15452, 2023b.

Rame, A., Dancette, C., and Cord, M. Fishr: Invariant gradient variances for out-of-distribution generalization. In *ICML*, pp. 18347–18377, 2022.

Ramrakhya, R., Undersander, E., Batra, D., and Das, A. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *CVPR*, pp. 5173–5183, 2022.

Raychaudhuri, S., Wani, S., Patel, S., Jain, U., and Chang, A. X. Language-aligned waypoint (law) supervision for vision-and-language navigation in continuous environments. In *EMNLP*, pp. 4018–4028, 2021.

Shi, Y., Seely, J., Torr, P., Siddharth, N., Hannun, A., Usunier, N., and Synnaeve, G. Gradient matching for domain generalization. In *ICLR*, 2021.

Shlens, J. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

Song, J., Lee, J., Kweon, I. S., and Choi, S. Ecotta: Memory-efficient continual test-time adaptation via self-distilled regularization. In *CVPR*, pp. 11920–11929, 2023.

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., and Hardt, M. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*, pp. 9229–9248, 2020.

Tan, H., Yu, L., and Bansal, M. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of NAACL-HLT*, pp. 2610–2621, 2019.

Tang, Y., Zhang, C., Xu, H., Chen, S., Cheng, J., Leng, L., Guo, Q., and He, Z. Neuro-modulated hebbian learning for fully test-time adaptation. In *CVPR*, pp. 3728–3738, 2023.

Tian, J., He, Z., Dai, X., Ma, C.-Y., Liu, Y.-C., and Kira, Z. Trainable projected gradient method for robust fine-tuning. In *CVPR*, pp. 7836–7845, 2023.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. In *ICLR*, 2021.

Wang, H., Liang, W., Gool, L. V., and Wang, W. Dreamwalker: Mental planning for continuous vision-language navigation. In *ICCV*, pp. 10873–10883, 2023a.

Wang, P., Zhang, Z., Lei, Z., and Zhang, L. Sharpness-aware gradient matching for domain generalization. In *CVPR*, pp. 3769–3778, 2023b.

Wang, Q., Fink, O., Van Gool, L., and Dai, D. Continual test-time domain adaptation. In *CVPR*, pp. 7201–7211, 2022a.

Wang, S., Montgomery, C., Orbay, J., Birodkar, V., Faust, A., Gur, I., Jaques, N., Waters, A., Baldridge, J., and Anderson, P. Less is more: Generating grounded navigation instructions from landmarks. In *CVPR*, pp. 15428–15438, 2022b.

Wang, T., Wu, Z., Yao, F., and Wang, D. Graph based environment representation for vision-and-language navigation in continuous environments. *ArXiv*, abs/2301.04352, 2023c.

Wang, X., Huang, Q., Celikyilmaz, A., Gao, J., Shen, D., Wang, Y.-F., Wang, W. Y., and Zhang, L. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *CVPR*, pp. 6629–6638, 2019.

Wang, X., Wang, W., Shao, J., and Yang, Y. Lana: A language-capable navigator for instruction following and generation. In *CVPR*, pp. 19048–19058, 2023d.

Wang, Z., Grigsby, J., and Qi, Y. Pgrad: Learning principal gradients for domain generalization. In *ICLR*, 2023e.

Wang, Z., Li, X., Yang, J., Liu, Y., and Jiang, S. Gridmm: Grid memory map for vision-and-language navigation. In *ICCV*, pp. 15625–15636, 2023f.

Yang, Z., Majumdar, A., and Lee, S. Behavioral analysis of vision-and-language navigation agents. In *CVPR*, pp. 2574–2582, 2023.

Yi, C., YANG, S., Wang, Y., Li, H., Tan, Y.-p., and Kot, A. Temporal coherent test time optimization for robust video classification. In *ICLR*, 2023.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. In *NeurIPS*, volume 33, pp. 5824–5836, 2020.

Yuan, L., Xie, B., and Li, S. Robust test-time adaptation in dynamic scenarios. In *CVPR*, pp. 15922–15932, 2023.

Zhang, M., Levine, S., and Finn, C. Memo: Test time robustness via adaptation and augmentation. In *NeurIPS*, pp. 38629–38642, 2022.

Zhao, B., Chen, C., and Xia, S.-T. Delta: Degradation-free fully test-time adaptation. In *ICLR*, 2023.

Zhu, F., Zhu, Y., Chang, X., and Liang, X. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *CVPR*, pp. 10012–10022, 2020.

Zhu, F., Liang, X., Zhu, Y., Yu, Q., Chang, X., and Liang, X. Soon: Scenario oriented object navigation with graph-based exploration. In *CVPR*, pp. 12689–12699, 2021.

A. Implementation Details

Online vision-and-language navigation refers to a VLN setting wherein an agent navigates through diverse environments, possessing the capability to continuously update and adapt its model upon encountering new test data. To date, the majority of existing VLN tasks have not adhered to an online setting. Typically, they follow a distinct train-test paradigm, wherein models are trained on a training set and then fixed and evaluated on a test set without undergoing adaptive model updates during testing. To better conform to real-world application scenarios, for all datasets, we set the batch size to 1 during evaluation. Each sample (or each action step) is forward propagated only once during the testing process. Note that for the base models, we report the results obtained from running their official codes. For VLN models equipped with TTA strategies, we run the corresponding experiments 5 times while shuffling the order of the samples and report the average results. In our FSTTA and other compared TTA strategies, we only utilize the last four LN layers of the base models for model updating, all the feature dimensions of these layers are 768. We set the intervals for fast and slow updates to $M = 3$ and $N = 4$. For the dynamic learning rate scaling, we empirically set the threshold $\tau = 0.7$ in Eq. (6) and the update momentum $\rho = 0.95$ with the truncation interval $[a, b] = [0.9, 1.1]$. And the hyper-parameter q in Eq. (8) is set to 0.1. The learning rates of the two phases are $\hat{\gamma}^{(fast)} = 6 \times 10^{-4}$ and $\gamma^{(slow)} = 1 \times 10^{-3}$ for base model DUET, whereas the learning rates with base model HM3D are $\hat{\gamma}^{(fast)} = 4 \times 10^{-4}$ and $\gamma^{(slow)} = 2 \times 10^{-3}$. In addition, due to the disparities between discrete and continuous environments, we employ specific hyper-parameters for the R2R-CE dataset. Specifically, for this dataset, the learning rates of the two phases are $\hat{\gamma}^{(fast)} = 5 \times 10^{-4}$ and $\gamma^{(slow)} = 1 \times 10^{-3}$. The intervals for fast and slow updates are set to $M = 7$ and $N = 4$. To speed up the SVD decomposition, we utilize the fast SVD approach in (Wang et al., 2023e). All experiments are conducted on a RTX 3090 GPU. Our model is implemented with PyTorch 1.7.1 and Python 3.8.5, required packages are listed in our code.

Table 9. Experimental results on REVERIE datasets. Results better than the base model are highlighted in bold.

Methods	**REVERIE** Val Seen						**REVERIE** Val Unseen						**REVERIE** Test Unseen					
	TL ↓	OSR	SR	SPL	RGS	RGSPL	TL ↓	OSR	SR	SPL	RGS	RGSPL	TL ↓	OSR	SR	SPL	RGS	RGSPL
Human	-	-	-	-	-	-	-	-	-	-	-	-	21.18	86.83	81.51	53.66	77.84	51.44
Seq2Seq (Anderson et al., 2018)	12.88	35.70	29.59	24.01	18.97	14.96	11.07	8.07	4.20	2.84	2.16	1.63	10.89	6.88	3.99	3.09	2.00	1.58
RCM (Wang et al., 2019)	10.70	29.44	23.33	21.82	16.23	15.36	11.98	14.23	9.29	6.97	4.89	3.89	10.60	11.68	7.84	6.67	3.67	3.14
SMNA (Ma et al., 2019a)	7.54	43.29	41.25	39.61	30.07	28.98	9.07	11.28	8.15	6.44	4.54	3.61	9.23	8.39	5.80	4.53	3.10	2.39
FAST (Qi et al., 2020)	16.35	55.17	50.53	45.50	31.97	29.66	45.28	28.20	14.40	7.19	7.84	4.67	39.05	30.63	19.88	11.61	11.28	6.08
ORIST (Qi et al., 2021)	10.73	49.12	45.19	42.21	29.87	27.77	10.90	25.02	16.84	15.14	8.52	7.58	11.38	29.20	22.19	18.97	10.68	9.28
CKR (Gao et al., 2021a)	12.16	61.91	57.27	53.57	39.07	-	26.26	31.44	19.14	11.84	11.45	-	22.46	30.40	22.00	14.25	11.60	-
SIA (Lin et al., 2021)	13.61	65.85	61.91	57.08	45.96	42.65	41.53	44.67	31.53	16.28	22.41	11.56	48.61	44.56	30.80	14.85	19.02	9.20
RecBERT (Hong et al., 2021)	13.44	53.90	51.79	47.96	38.23	35.61	16.78	35.02	30.67	24.90	18.77	15.27	15.86	32.91	29.61	23.99	16.50	13.51
Airbert (Guhur et al., 2021)	15.16	48.98	47.01	42.34	32.75	30.01	18.71	34.51	27.89	21.88	18.23	14.18	17.91	34.20	30.28	23.61	16.83	13.28
HAMT (Chen et al., 2021)	12.79	47.65	43.29	40.19	27.20	25.18	14.08	36.84	32.95	30.20	18.92	17.28	13.62	33.41	30.40	26.67	14.88	13.08
HOP (Qiao et al., 2022)	13.80	53.76	54.88	47.19	38.65	33.85	16.46	36.24	31.78	26.11	18.85	15.73	16.38	33.06	30.17	24.34	17.69	14.34
BEVBert (An et al., 2022)	-	76.18	73.72	65.32	57.70	51.73	-	56.40	51.78	36.37	34.71	24.44	-	57.26	52.81	36.41	32.06	22.09
MiC (Qiao et al., 2023a)	-	-	-	-	-	-	20.64	62.37	56.97	43.60	37.52	28.72	18.11	62.40	55.74	41.97	35.25	26.17
LANA (Wang et al., 2023d)	15.91	74.28	71.94	62.77	59.02	50.34	23.18	52.97	48.31	33.86	32.86	22.77	18.83	57.20	51.72	36.45	32.95	22.85
BSG (Liu et al., 2023)	15.26	78.36	76.18	66.69	61.56	54.02	24.71	58.05	52.12	35.59	35.36	24.24	22.90	62.83	56.45	38.70	33.15	22.34
GridMM (Wang et al., 2023f)	-	-	-	-	-	-	23.20	57.48	51.37	36.47	34.57	24.56	19.97	59.55	55.13	36.60	34.87	23.45
DUET (Chen et al., 2022d)	13.86	73.86	71.75	63.94	57.41	51.14	22.11	51.07	46.98	33.73	32.15	23.03	21.30	56.91	52.51	36.06	31.88	22.06
DUET-FSTTA	15.13 ± 0.13	**75.59 ± 0.25**	**75.48 ± 0.19**	**65.84 ± 0.07**	**58.62 ± 0.43**	**52.23 ± 0.18**	22.14 ± 0.22	**56.26 ± 0.14**	**54.15 ± 0.19**	**36.41 ± 0.13**	**34.27 ± 0.56**	**23.56 ± 0.30**	21.52 ± 0.10	**58.44 ± 0.13**	**53.40 ± 0.37**	**36.43 ± 0.23**	**32.99 ± 0.43**	**22.40 ± 0.16**
HM3D (Chen et al., 2022c)	16.18	66.76	65.00	55.70	48.42	41.66	22.13	62.11	55.89	40.85	36.58	26.76	20.87	59.81	53.13	38.24	32.69	22.68
HM3D-FSTTA	16.24 ± 0.10	**69.41 ± 0.13**	**67.79 ± 0.27**	**58.28 ± 0.23**	48.14 ± 0.46	41.60 ± 0.18	22.37 ± 0.39	**63.74 ± 0.05**	**57.02 ± 0.21**	**41.41 ± 0.30**	**36.97 ± 0.37**	26.55 ± 0.13	21.90 ± 0.14	**63.68 ± 0.07**	**56.44 ± 0.08**	**39.58 ± 0.27**	**34.05 ± 0.66**	**23.04 ± 0.12**

B. Complementary Experiments

B.1. Full Results on the 4 Benchmarks

In our main paper, due to the space limitation, we only provide the representative comparison results on REVERIE (Qi et al., 2020), SOON (Zhu et al., 2021), R2R (Anderson et al., 2018), and R2R-CE (Krantz et al., 2020) benchmarks. Here, we show the full results on the 'validation seen', 'validation unseen ', and 'test unseen' splits of these benchmarks by comparing more state-of-the-art methods. Note that since the base models DUET (Chen et al., 2022d) and HM3D (Chen et al., 2022c) produce errors when online evaluating on the R2R 'test unseen' set, we don't perform evaluation on this split. Moreover, we follow (Chen et al., 2022d) to use the challenge splits of SOON for evaluation[1]. Table 9, Table 10, Table 11, and Table 12 show the full comparison with more approaches. Since the results of our proposed FSTTA are averaged over 5 random runs, we report the mean and standard deviation. To further enhance the performance of online VLN, we will introduce additional strategies for open-environment sensing (Hu et al., 2024; Gao et al., 2023b) and entity localization (Gao & Xu, 2021) in the future, enabling the agent to acquire more comprehensive information. In addition, introducing external knowledge may be a viable path to enhancing online VLN performance (Gao et al., 2021b).

In addition, to verify that our proposed FSTTA can handle larger domain gap, we design a new experimental setting that

[1]As shown in `https://github.com/ZhuFengdaaa/SOON/issues/1`, The SOON dataset (Zhu et al., 2021) does not release the split in their original paper. Therefore, performance comparisons are based on their challenge report `https://scenario-oriented-objectnavigation.github.io`.

Table 10. Experimental results on R2R datasets.

Methods	**R2R** Val Seen					**R2R** Val Unseen				
	TL ↓	NE ↓	OSR	SR	SPL	TL ↓	NE ↓	OSR	SR	SPL
Seq2Seq (Anderson et al., 2018)	11.33	6.01	53	39	-	8.39	7.81	28	22	-
SF (Fried et al., 2018)	-	3.36	74	66	-	-	6.62	45	35	-
Chasing (Anderson et al., 2019)	10.15	7.59	42	34	30	9.64	7.20	44	35	31
RCM (Wang et al., 2019)	10.65	3.53	75	67	-	11.46	6.09	50	43	-
SMNA (Ma et al., 2019a)	-	3.22	78	67	58	-	5.52	56	45	32
PRESS (Li et al., 2019)	10.57	4.39	-	58	55	10.36	5.28	-	49	45
EnvDrop (Tan et al., 2019)	11.00	3.99	-	62	59	10.70	5.22	-	52	48
AuxRN (Zhu et al., 2020)	-	3.33	78	70	67	-	5.28	62	55	50
PREVALENT (Hao et al., 2020)	10.32	3.67	-	69	65	10.19	4.71	-	58	53
RelGraph (Hong et al., 2020)	10.13	3.47	-	67	65	9.99	4.73	-	57	53
NvEM (An et al., 2021)	11.09	3.44	-	69	65	11.83	4.27	-	60	55
RecBERT (Hong et al., 2021)	11.13	2.90	-	72	68	12.01	3.93	-	63	57
AirBert (Guhur et al., 2021)	11.09	2.68	-	75	70	11.78	4.10	-	62	56
REM (Liu et al., 2021)	10.88	2.48	-	75	72	12.44	3.89	-	64	58
HAMT (Chen et al., 2021)	11.15	2.51	-	76	72	11.46	3.65	-	66	61
GBE (Zhu et al., 2021)	-	-	-	-	-	-	5.20	-	54	43
SEvol (Chen et al., 2022a)	11.97	3.56	-	67	63	12.26	3.99	-	62	57
HOP (Qiao et al., 2022)	11.26	2.72	-	75	70	12.27	3.80	-	64	57
DAVIS (Lu et al., 2022)	12.45	3.16	-	80	76	12.65	3.16	-	67	61
BEVBert (An et al., 2022)	13.56	2.17	88	81	74	14.55	2.81	84	75	64
LANA (Wang et al., 2023d)	-	-	-	-	-	12.00	-	76	68	62
BSG (Liu et al., 2023)	-	-	-	-	-	14.90	2.89	-	74	62
GridMM (Wang et al., 2023f)	-	-	-	-	-	13.27	2.83	-	75	64
DUET (Chen et al., 2022d)	12.33	2.28	86	79	73	13.94	3.31	81	72	60
DUET-FSTTA	13.39 ± 0.14	**2.25** ± **0.29**	**86** ± **0.2**	**79** ± **0.2**	**73** ± **0.2**	14.64 ± 0.12	**3.03** ± **0.22**	**82** ± **0.2**	**75** ± **0.6**	**62** ± **0.2**
HM3D (Chen et al., 2022c)	13.30	2.70	84	77	71	14.29	2.83	83	74	62
HM3D-FSTTA	13.52 ± 0.12	**2.57** ± **0.39**	**85** ± **0.2**	**78** ± **0.4**	**71** ± **0.2**	14.86 ± 0.10	**2.71** ± **0.48**	82 ± 0.2	**75** ± **0.4**	**63** ± **0.2**

Table 11. Experimental results on SOON datasets.

Methods	**SOON** Val Unseen					**SOON** Test Unseen				
	TL ↓	OSR	SR	SPL	RGSPL	TL ↓	OSR	SR	SPL	RGSPL
GBE (Zhu et al., 2021)	28.96	28.54	19.52	13.34	1.16	27.88	21.45	12.90	9.23	0.45
GridMM (Wang et al., 2023f)	38.92	53.39	37.46	24.81	3.91	46.20	48.02	36.27	21.25	4.15
KERM (Li et al., 2023)	35.83	51.62	38.05	23.16	4.04	-	-	-	-	-
AZHP (Gao et al., 2023a)	39.33	56.19	40.71	26.58	5.53	-	-	-	-	-
DUET (Chen et al., 2022d)	36.20	50.91	36.28	22.58	3.75	41.83	43.00	33.44	21.42	4.17
DUET-FSTTA	**35.32** ± **0.51**	**52.57** ± **0.06**	**36.53** ± **0.26**	**23.82** ± **0.21**	**3.75** ± **0.02**	41.93 ± 0.25	**43.44** ± **0.21**	**35.34** ± **0.18**	**23.23** ± **0.48**	**4.52** ± **0.04**
HM3D (Chen et al., 2022c)	34.13	53.22	41.00	30.69	4.06	36.64	47.26	40.26	28.09	5.15
HM3D-FSTTA	34.91 ± 0.22	**54.19** ± **0.20**	**42.44** ± **0.44**	**31.03** ± **0.34**	**4.93** ± **0.09**	37.52 ± 0.12	**48.52** ± **0.26**	**42.02** ± **0.24**	**28.95** ± **0.45**	**5.20** ± **0.05**

transfers models trained on discrete dataset to continuous dataset. Here, the substantial gap between discrete and continuous environments poses a challenge for the effective TTA. As shown in Table 12, DUET$_{discrete}$ refers to the DUET model pretrained on discrete R2R dataset without finetuning on the continuous data. We can observe that the significant domain discrepancy between discrete and continuous navigation environments leads to a serious performance drop for agents. Experimenal results show that our FSTTA method effectively assists the model in enhancing its generalization ability.

Comparison of Time Costs. In our framework, using TTA inevitably incurs time overhead. Here, we compare the time costs with several state-of-the-art methods. From Table 13 we can observe that our proposed approaches (DUET-FSTTA, HM3D-FSTTA) achieve significant performance with moderate time cost. Note that, some methods, such as GridMM and BEVBert, which use the features of bird-eye-view, have higher computational cost than our TTA-equipped methods.

B.2. More Results for Comparison with Other TTA Strategies

In our main paper, six state-of-the-art TTA strategies are adopted for comparison including Tent (Wang et al., 2021), EATA (Niu et al., 2022), CoTTA (Wang et al., 2022a), NOTE (Gong et al., 2022), SAR (Niu et al., 2023) and ViDA (Liu et al., 2024). The settings of these TTA strategies for VLN are introduced as follows:

Tent (Wang et al., 2021). We follow all hyper-parameters that are set in Tent. The optimizer is AdamW (Loshchilov & Hutter, 2017) and the learning rate for batch size = 1 is set to (0.001/64).

EATA (Niu et al., 2022). We follow all hyper-parameters that are set in EATA. Specifically, the entropy constant E_0 (for reliable sample identification) is set to $0.1 \times \ln 1000$. The ϵ for redundant sample identification is set to 0.05. The trade-off parameter β for entropy loss and regularization loss is set to 2000. The number of pre-collected in-distribution test samples for Fisher importance calculation is 2,000. The update rule is the same as Tent.

CoTTA (Wang et al., 2022a). We follow all hyper-parameters that are set in CoTTA. Specifically, we use random augmentation compositions including gaussian noise and dropout for our experiments, with a confidence threshold of 0.55. The same AdamW optimizer as Tent is utilized for the practical implementation. The restoration probability is set to 0.01

Table 12. Experimental results on R2R-CE datasets.

Methods	**R2R-CE** Val Seen					**R2R-CE** Val Unseen					**R2R-CE** Test Unseen				
	TL ↓	NE ↓	OSR	SR	SPL	TL ↓	NE ↓	OSR	SR	SPL	TL ↓	NE ↓	OSR	SR	SPL
Seq2Seq (Krantz et al., 2020)	9.26	7.12	46	37	35	8.64	7.37	40	32	30	8.85	7.91	36	28	25
SASRA (Irshad et al., 2021)	8.89	7.71	-	36	34	7.89	8.32	-	24	22	-	-	-	-	-
CWTP (Chen et al., 2020)	-	7.10	56	36	31	-	7.90	38	26	23	-	-	-	-	-
AG-CMTP (Chen et al., 2022a)	-	6.60	56.2	35.9	30.5	-	7.90	39.2	23.1	19.1	-	-	-	-	-
R2R-CMTP (Chen et al., 2022a)	-	7.10	45.4	36.1	31.2	-	7.90	38.0	26.4	22.7	-	-	-	-	-
LAW (Raychaudhuri et al., 2021)	9.34	6.35	49	40	37	8.89	6.83	44	35	31	9.67	7.69	28	38	25
WPN (Krantz et al., 2021)	8.54	5.48	53	46	43	7.62	6.31	40	36	34	8.02	6.65	37	32	30
CM^2 (Georgakis et al., 2022)	12.05	6.10	50.7	42.9	34.8	11.54	7.02	42	34	28	13.90	7.70	39	31	24
CM^2-GT (Georgakis et al., 2022)	12.60	4.81	58.3	52.8	41.8	10.68	6.23	41	37	31	-	-	-	-	-
WS-MGMap (Chen et al., 2022b)	10.12	5.65	51.7	46.9	43.4	10.00	6.28	48	39	34	12.30	7.11	45	35	28
Sim2Sim (Krantz & Lee, 2022)	11.18	4.67	61	46	42	10.69	6.07	52	43	36	11.43	6.17	52	44	37
CWP-CMA (Hong et al., 2022)	11.47	5.20	61	51	45	10.90	6.20	52	41	36	11.85	6.30	49	38	33
CWP-BERT (Hong et al., 2022)	12.50	5.02	59	50	44	12.23	5.74	53	44	39	13.51	5.89	51	42	36
ERG (Wang et al., 2023c)	11.8	5.04	61	46	42	9.96	6.20	52	41	36	-	-	-	-	-
DREAMW (Wang et al., 2023a)	11.6	4.09	59	66	48	11.3	5.53	49	59	44	11.8	5.48	49	57	44
GridMM (Wang et al., 2023f)	12.69	4.21	69	59	51	13.36	5.11	61	49	41	13.31	5.64	56	46	39
ETPNav (An et al., 2023)	11.78	3.95	72	66	59	11.99	4.71	65	57	49	12.87	5.12	63	55	48
DUET (Chen et al., 2022d)	12.62	4.13	67	57	49	11.86	5.13	55	46	40	13.13	5.82	50	42	36
DUET-FSTTA	**12.39** ± **0.30**	4.25 ± 0.23	**69** ± **0.6**	**58**± **0.2**	**50**± **0.0**	11.58 ± 0.17	5.27 ± 0.12	**58** ± **0.4**	**48** ± **0.6**	**42** ± **0.4**	13.17 ± 0.10	5.84 ± 0.22	**55** ± **0.8**	**46** ± **0.2**	**38** ± **0.2**
DUET$_{discrete}$ [*]	10.34	5.18	64	55	48	9.97	6.47	53	47	38	11.45	7.01	47	40	35
DUET$_{discrete}$-FSTTA	**10.25** ± **0.47**	5.22 ± 0.29	**65** ± **0.4**	**56**± **0.3**	**49**± **0.2**	10.16 ± 0.28	**5.61** ± **0.19**	**55** ± **0.2**	**48** ± **0.4**	**38** ± **0.2**	11.85 ± 0.37	**6.45** ± **0.31**	**48** ± **0.4**	**43** ± **0.5**	**36** ± **0.7**
BEVBert (An et al., 2022)	13.98	3.77	73	68	60	13.27	4.57	67	59	50	15.31	4.70	67	59	50
BEVBert-FSTTA	14.07 ± 0.49	4.11 ± 0.22	**74** ± **0.4**	**69** ± **0.4**	**60** ± **0.6**	**13.11** ± **0.46**	**4.39** ± **0.22**	65 ± 0.6	**60** ± **0.4**	**51** ± **0.6**	**15.02** ± **0.51**	5.45 ± 0.31	**69** ± **0.4**	**60** ± **0.2**	**50** ± **0.4**

[*] Note that DUET$_{discrete}$ refers to the DUET model pretrained on discrete R2R dataset without finetuning on the continuous data. In this setting, a significant domain discrepancy between discrete and continuous environments exists.

Table 13. Experimental results for different methods with time costs on REVERIE dataset.

Methods	**REVERIE** Val Unseen						Time(ms)[*]
	TL ↓	OSR	SR	SPL	RGS	RGSPL	
DUET (Chen et al., 2022d)	22.11	51.07	46.98	33.73	32.15	23.03	104.84
+ EATA (Niu et al., 2022)	23.41	52.09	47.40	33.46	32.09	22.65	133.12
+ CoTTA (Wang et al., 2022a)	24.88	52.46	47.56	31.43	31.82	21.83	3.89×10^3
+ NOTE (Gong et al., 2022)	23.15	52.85	48.28	33.98	32.77	22.98	137.89
+ SAR (Niu et al., 2023)	23.47	53.26	48.00	33.92	33.49	23.09	145.53
+ ViDA (Liu et al., 2024)	24.53	52.53	48.14	32.45	32.26	21.92	3.97×10^3
+ Tent (Wang et al., 2021)	24.05	49.43	46.87	31.90	30.04	20.15	126.91
+ Tent-INT-2	24.24	51.22	48.46	33.67	32.43	21.30	124.02
+ Tent-INT-3	22.52	52.28	48.60	34.65	32.66	23.12	119.34
+ Tent-INT-4	22.59	51.40	48.91	35.06	32.59	22.99	117.26
+ Tent-Stable	**22.05**	51.43	47.55	33.99	32.34	23.32	129.22
+ FSTTA	22.14	**56.26**	**54.15**	**36.41**	**34.27**	**23.56**	135.61
HM3D (Chen et al., 2022c)	22.13	62.11	55.89	40.85	36.58	26.76	107.96
+FSTTA	22.37	**63.74**	**57.02**	**41.41**	**36.97**	26.55	141.70
BEVBert (An et al., 2022)	-	56.40	51.78	36.37	34.71	24.44	161.43
GridMM (Wang et al., 2023f)	23.20	57.48	51.37	36.47	34.57	24.56	277.44

[*] Note that the last column displays the average execution time of the agent for a single instruction, calculated on the Validation Unseen set of REVERIE.

and EMA factor is set to 0.999.

NOTE (Gong et al., 2022). We follow all hyper-parameters that are set in NOTE. Specifically, the soft-shrinkage width is set to 4 and EMA momentum is set to 0.01. The update rule is AdamW with a learning rate as 0.0001.

SAR (Niu et al., 2023). We follow all hyper-parameters that are set in SAR. Specifically, the entropy constant E_0 (for reliable sample identification) is set to $0.4 \times \ln 1000$, and the neighborhood size for sharpness-aware minimization is set by the default value 0.05. For model recovery, the moving average factor is set to 0.9 and the reset threshold is set to 0.2.

ViDA (Liu et al., 2024). We follow all hyper-parameters that are set in ViDA. Specifically, we use random augmentation compositions including gaussian noise and dropout for our experiments. The same AdamW optimizer as Tent is utilized for the practical implementation. The threshold value is set to 0.2 and the updating weight is set to 0.999.

B.2.1. Detailed Comparison with TTA Approaches

In our main paper, we compare the proposed FSTTA approach with various TTA method on the REVERIE dataset. Here we provide detailed comparison with the mean and standard deviation. From Table 14 we can find that our proposed FSTTA obtains more stable results (with lower standard deviation).

Table 14. Experimental results for different TTA strategies.

Methods	**REVERIE** Val Seen						**REVERIE** Val Unseen						**REVERIE** Test Unseen					
	TL ↓	OSR	SR	SPL	RGS	RGSPL	TL ↓	OSR	SR	SPL	RGS	RGSPL	TL ↓	OSR	SR	SPL	RGS	RGSPL
DUET (Chen et al., 2022d)	13.86	73.86	71.75	63.94	57.41	51.14	22.11	51.07	46.98	33.73	32.15	23.03	21.30	56.91	52.51	36.06	31.88	22.06
+ EATA (Niu et al., 2022)	14.48 ± 0.49	74.05 ± 0.93	72.17 ± 1.85	63.41 ± 1.55	56.92 ± 0.88	49.94 ± 0.65	23.41 ± 0.63	52.09 ± 0.61	47.40 ± 1.13	33.46 ± 1.47	32.09 ± 1.43	22.65 ± 0.33	22.15 ± 0.50	57.21 ± 0.72	52.91 ± 1.20	35.19 ± 1.39	31.39 ± 1.30	21.65 ± 0.29
+ CoTTA (Wang et al., 2022a)	15.75 ± 1.62	73.79 ± 1.21	71.05 ± 1.19	62.91 ± 0.86	55.64 ± 1.72	49.36 ± 1.21	24.88 ± 2.01	52.46 ± 1.22	47.56 ± 1.10	31.43 ± 1.03	31.82 ± 1.66	21.83 ± 1.55	23.45 ± 2.20	56.14 ± 1.96	52.52 ± 1.18	34.66 ± 1.34	31.28 ± 0.97	21.06 ± 1.04
+ NOTE (Gong et al., 2022)	14.32 ± 0.22	74.76 ± 0.77	72.43 ± 0.51	64.28 ± 0.60	58.61 ± 0.93	51.97 ± 0.55	23.15 ± 0.12	52.85 ± 0.20	48.28 ± 1.24	33.98 ± 0.96	32.77 ± 0.69	22.98 ± 0.65	21.85 ± 0.28	57.66 ± 0.36	53.41 ± 0.94	36.18 ± 0.86	32.21 ± 0.72	22.09 ± 0.43
+ SAR (Niu et al., 2023)	14.32 ± 0.66	74.84 ± 1.79	71.75 ± 1.27	64.43 ± 0.49	57.70 ± 0.60	51.70 ± 0.15	23.47 ± 0.70	53.26 ± 1.52	48.00 ± 1.75	33.92 ± 0.50	33.49 ± 1.14	23.09 ± 0.55	21.56 ± 0.99	57.11 ± 1.51	53.04 ± 0.98	36.07 ± 0.30	32.64 ± 0.20	22.27 ± 0.87
+ ViDA (Liu et al., 2024)	15.25 ± 1.02	73.99 ± 0.87	72.49 ± 0.83	63.49 ± 0.70	57.20 ± 1.23	50.89 ± 0.63	24.53 ± 0.97	52.53 ± 0.29	48.14 ± 1.10	32.45 ± 0.98	32.26 ± 1.54	21.92 ± 0.79	22.74 ± 1.05	56.78 ± 1.47	52.74 ± 1.16	35.10 ± 0.35	31.89 ± 0.41	21.77 ± 0.66
+ Tent (Wang et al., 2021)	15.51 ± 0.87	70.07 ± 1.03	70.73 ± 0.49	61.67 ± 1.31	55.19 ± 0.77	49.31 ± 0.51	24.05 ± 1.62	49.43 ± 1.25	46.87 ± 0.51	31.90 ± 1.53	30.04 ± 0.66	20.15 ± 0.35	23.88 ± 1.35	54.58 ± 1.41	50.56 ± 0.46	33.37 ± 0.79	30.30 ± 1.23	20.32 ± 0.42
+ Tent-INT-2	14.97 ± 0.45	71.82 ± 0.87	70.36 ± 0.42	61.53 ± 0.96	54.88 ± 0.54	49.82 ± 0.21	24.24 ± 1.84	51.22 ± 1.30	48.46 ± 0.48	33.67 ± 1.93	32.43 ± 0.36	21.30 ± 0.62	23.19 ± 1.02	54.25 ± 1.41	50.36 ± 0.67	33.89 ± 0.32	30.30 ± 0.71	21.09 ± 0.29
+ Tent-INT-3	14.18 ± 0.60	74.54 ± 0.89	72.58 ± 0.38	64.44 ± 1.02	56.96 ± 0.96	51.05 ± 0.27	22.52 ± 0.96	52.28 ± 0.70	48.60 ± 0.33	34.65 ± 0.84	32.66 ± 0.81	23.12 ± 0.28	21.95 ± 0.95	56.66 ± 0.62	52.92 ± 0.64	35.84 ± 0.18	31.98 ± 0.93	21.89 ± 0.37
+ Tent-INT-4	14.12 ± 0.51	73.84 ± 0.98	72.51 ± 0.60	64.03 ± 0.20	56.70 ± 0.93	50.97 ± 0.59	22.59 ± 0.80	51.40 ± 0.24	48.91 ± 0.40	35.06 ± 0.67	32.59 ± 0.36	22.99 ± 0.28	21.45 ± 0.61	56.74 ± 0.22	53.14 ± 0.42	36.29 ± 0.10	32.11 ± 0.62	22.14 ± 0.29
+ Tent-Stable	13.90 ± 0.74	73.72 ± 1.33	71.89 ± 0.29	64.06 ± 0.52	56.43 ± 0.95	50.41 ± 0.44	22.05 ± 0.95	51.43 ± 1.18	47.55 ± 0.39	33.99 ± 1.06	32.34 ± 0.55	23.32 ± 0.70	21.15 ± 0.88	57.12 ± 1.31	52.61 ± 1.04	36.17 ± 0.82	32.15 ± 0.76	22.16 ± 0.34
+ FSTTA	15.13 ± 0.13	**75.59 ± 0.25**	**75.48 ± 0.19**	**65.84 ± 0.07**	**61.69 ± 0.43**	**52.23 ± 0.18**	22.14 ± 0.22	**56.26 ± 0.14**	**54.15 ± 0.19**	**36.41 ± 0.13**	**34.27 ± 0.56**	**23.56 ± 0.30**	21.52 ± 0.10	**58.44 ± 0.13**	**53.40 ± 0.37**	**36.43 ± 0.23**	**32.99 ± 0.43**	**22.40 ± 0.16**

Table 15. Experimental results for different TTA strategies with their **default** updated parameters.

Methods	**REVERIE** Val Unseen					
	TL ↓	OSR	SR	SPL	RGS	RGSPL
DUET (Chen et al., 2022d)	**22.11**	51.07	46.98	33.73	32.15	23.03
+ Tent (Wang et al., 2021)	23.71 ± 1.95	48.87 ± 1.65	44.28 ± 1.84	28.88 ± 0.94	30.70 ± 0.86	19.89 ± 0.73
+ EATA (Niu et al., 2022)	22.06 ± 1.62	52.85 ± 1.52	45.44 ± 0.89	32.79 ± 0.94	32.13 ± 1.27	23.14 ± 0.53
+ CoTTA (Wang et al., 2022a)	27.13 ± 2.88	43.48 ± 3.46	39.25 ± 2.56	25.44 ± 1.31	27.15 ± 0.94	17.01 ± 1.03
+ NOTE (Gong et al., 2022)	22.84 ± 1.85	50.78 ± 1.94	45.90 ± 0.71	31.82 ± 0.32	31.50 ± 0.66	22.14 ± 0.74
+ SAR (Niu et al., 2023)	22.26 ± 1.47	50.46 ± 1.85	44.39 ± 1.59	30.35 ± 0.91	31.03 ± 0.82	21.91 ± 0.56
+ FSTTA	22.14 ± 0.22	**56.26 ± 0.14**	**54.15 ± 0.19**	**36.41 ± 0.13**	**34.27 ± 0.56**	**23.56 ± 0.30**

B.2.2. Comparison with TTA Strategies that Update Their Default Number of Parameters

In the previous section and § 4.1 of our main paper, to make the comparison between different TTA approaches fair, we use the same amount of parameters for model updates among these methods. Astute readers might recognize that some of these strategies conventionally encompass a vast subset of model parameters, e.g., all normalization layers (Wang et al., 2021; Niu et al., 2022; Gong et al., 2022; Niu et al., 2023) or even the entirety of parameters (Wang et al., 2022a), for updates. Without these settings, the compared TTA strategies may not obtain promising perfromance. Consequently, we let these competitors use their default number of parameters for model update. However, the results in Table 15 indicate that even though these TTA methods update more parameters, they still fail to achieve better performance on the VLN task. Additionally, updating more parameters results in heavier computational burden. These findings underscore the challenges involved in devising effective TTA strategies for the online VLN task.

B.2.3. Comprehensive Comparison with Frequent and Stable Update Strategies

In Figure 1(b), we provide the performance comparison with various frequent and stable update strategies on SR and SLP metrics. To make a more comprehensive evaluation, as shown in Figure 4, we show the comparison results on all the metrics. Note that, 'Freq' means frequent update that updates at certain intervals within each sample. 'Stable' refers to stable update that initializes with the original base model (DUET) for each sample and uses different intra-sample update interval. 'INT' indicates the intra-sample update interval, which means averaging the gradient information over a certain action interval and then performing an iteration of model update. The results still reveal that both overly fast and overly slow model updates fail to achieve significant performance improvements, while our proposed FSTTA can achieve a better balance between adaptablility and stability for pursing favorable performance.

B.3. Parameter Analysis

In our method, there exist four important hyper-parameters: the step size and learning rate for fast and slow update, namely M, N, $\hat{\gamma}^{(fast)}$, and $\gamma^{(slow)}$, which controls the frequency and speed of model adaptation. Figure 5 and Figure 6 show that a moderate value of these parameters achieves favorable performance.

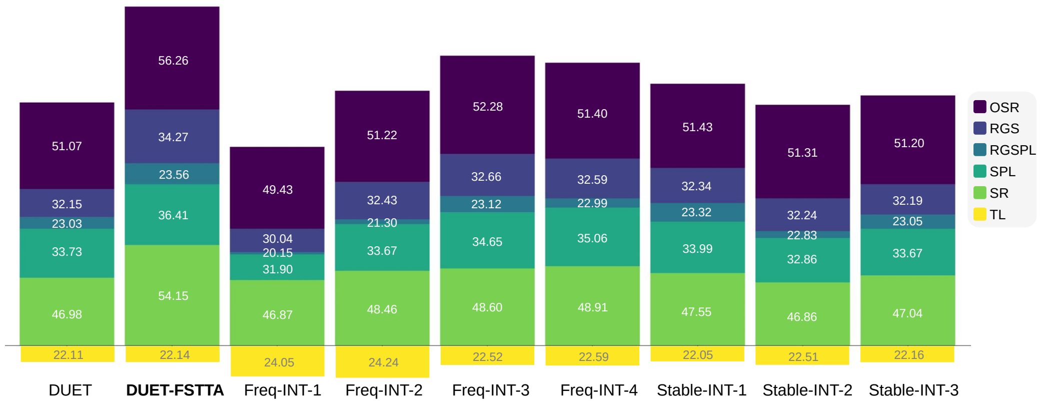
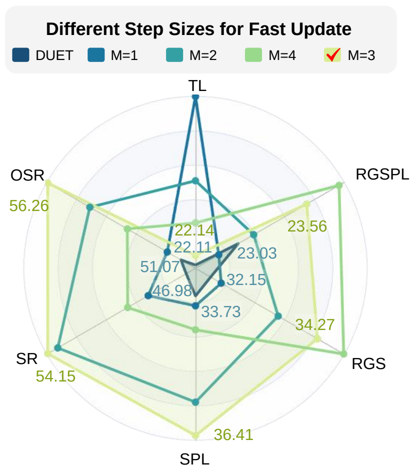


Figure 4. Experimental Results of Frequent and Stable Update Strategies on REVERIE 'Val Unseen' set. The detailed performance value for each methods are indicated.



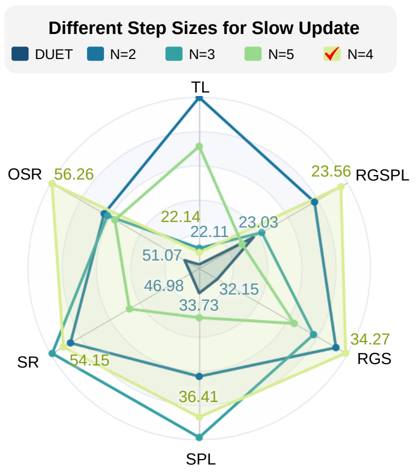
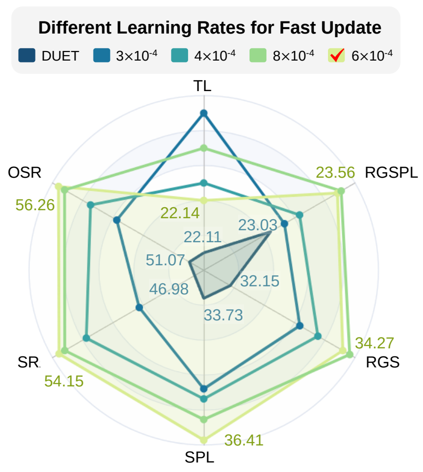


Figure 5. Experimental results for different settings of step size in fast update phase (*left*) and slow update phase (*right*).



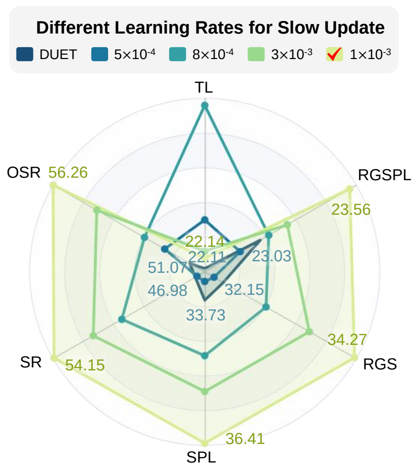


Figure 6. Experimental results for different settings of learning rate in fast update phase (*left*) and slow update phase (*right*).