Search-TTA: A Multimodal Test-Time Adaptation Framework for Visual Search in the Wild

Project Page: `https://search-tta.github.io`

Derek Ming Siang Tan[1,4†] **Shailesh**[3] **Boyang Liu**[1] **Alok Raj**[3] **Qi Xuan Ang**[4]
Weiheng Dai[1] **Tanishq Duhan**[1] **Jimmy Chiun**[1] **Yuhong Cao**[1†]
Florian Shkurti[2] **Guillaume Sartoretti**[1]
[1]National University of Singapore [2]University of Toronto
[3]IIT-Dhanbad [4]Singapore Technologies Engineering

Abstract:

To perform outdoor autonomous visual navigation and search, a robot may leverage satellite imagery as a prior map. This can help inform high-level search and exploration strategies, even when such images lack sufficient resolution to allow for visual recognition of targets. However, there are limited training datasets of satellite images with annotated targets that are not directly visible. Furthermore, approaches which leverage large Vision Language Models (VLMs) for generalization may yield inaccurate outputs due to hallucination, leading to inefficient search. To address these challenges, we introduce **Search-TTA**, a multimodal test-time adaptation framework with a flexible plug-and-play interface compatible with various input modalities (e.g. image, text, sound) and planning methods. First, we pretrain a satellite image encoder to align with CLIP's visual encoder to output probability distributions of target presence used for visual search. Second, our framework dynamically refines CLIP's predictions during search using a test-time adaptation mechanism. Through a novel feedback loop inspired by Spatial Poisson Point Processes, uncertainty-weighted gradient updates are used to correct (potentially inaccurate) predictions and improve search performance. To train and evaluate Search-TTA, we curate **AVS-Bench**, a visual search dataset based on internet-scale ecological data that contains up to 380k training and 8k validation images (*in-* and *out-domain*). We find that Search-TTA improves planner performance and score map distribution by up to 30.0% and 8.5% respectively, particularly in cases with poor initial CLIP predictions due to limited training data. It also achieves zero-shot generalization to unseen modalities. Finally, we deploy Search-TTA on a real UAV via hardware-in-the-loop testing, by simulating its operation within a large-scale simulation that provides onboard sensing.

Keywords: Test Time Adaptation, VLN, Visual Search, Ecological Monitoring

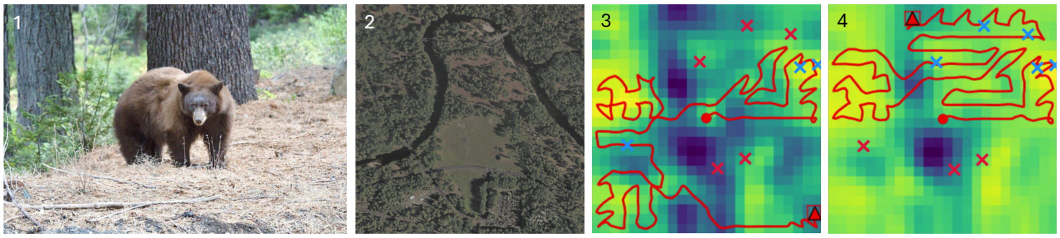


Figure 1: **Visual search for bears by a simulated UAV over Yosemite Valley (2).** (3) Utilizing a poor probability map leads to suboptimal search performance. (4) Test-time adaptation refines the target probability map by incorporating onboard measurements collected during search, guiding the UAV toward denser and less populated vegetation where bears are more likely to be found [1] [2].

[†]Correspondence to `derektan@u.nus.edu`, `caoyuhong@u.nus.edu`

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

1 Introduction

Recent advances in visual navigation have leveraged either purely vision-based approaches [3–6] or vision–language foundation models (VLN) [7–9] to achieve strong performance and generalization in real-world scenarios. VLN approaches are used in Object Navigation (*ObjectNav*) [10–12] tasks, where robots search for specific household objects in indoor environments. More broadly, Autonomous Visual Search (AVS) extends to outdoor settings, where robots actively explore natural environments to locate targets of interest, with applications in environmental monitoring [13], and search and rescue [14, 15]. Outdoor AVS can be particularly challenging due to limited battery life and sensor field-of-view (FOV). There, one strategy is to extract useful visual cues from coarser satellite images to direct the search process at a high level, even if the targets cannot be directly seen in these images. Most AVS approaches comprise a vision module and a search module. The vision module is responsible for processing visual semantics from satellite images and outputs useful likelihood information (i.e., a prior), either in the form of probability distributions [16] or visual embeddings [17]. The planner can then use these inputs to guide the agent towards areas with a higher likelihood of seeing targets, by taking measurements using its (higher-resolution) sensor.

There are several challenges with generating useful visual priors to guide the search module. First, there are limited *in-the-wild* datasets of satellite images with diverse annotated targets that are not directly visible. Even if such training data were available, conventional vision models trained on a narrow set of classes [16–18] may lack the capacity to reason beyond what is directly observable [19]. Vision-language models (VLMs) [20–22], pretrained on a large corpus of internet-scale data, provide a promising solution to this problem due to their advanced reasoning and generalization abilities [22–24]. Unlike conventional vision models, they can reason about correlations between target and semantics of the environment. Nevertheless, even the best VLMs may generate inaccurate visual outputs (or '*hallucinate*') due to insufficient/inaccurate training data [25] or when encountering inputs (i.e. satellite images, taxonomies) that are out-of-domain [26]. Over time, these inaccurate predictions persist as VLMs lack a mechanism to correct these errors during search [26].

To address these issues, we present **Search-TTA**, a multimodal test-time adaptation framework that refines a VLM's (potentially inaccurate) predictions online, using the agent's measurements during AVS. In this work, we use CLIP [20] as our lightweight VLM, and first align a satellite image encoder to the same representation space [27–29] as a vision encoder through patch-level contrastive learning. This enables the satellite image encoder to generate a score map by taking the cosine similarity between its per-patch embeddings and the embeddings of other modalities (e.g., ground image, text, sound). We then introduce a novel test-time adaptation feedback mechanism to refine CLIP's predictions based on new measurements. To achieve this, we take inspiration from Spatial Poisson Point Processes [30] to perform gradient updates to the satellite image encoder based on past measurements. We also enhance the loss function with an uncertainty-driven weighting scheme that acts as a regularizer to ensure stable gradient updates. To train and evaluate Search-TTA, we curate **AVS-Bench**, a visual search dataset based on internet-scale ecological data [31] comprising satellite images, each with targets and their corresponding ground-level image and taxonomic label (some with sound data). It contains up to **380k** training and **8k** validation images (*in-* and *out-domain*).

Search-TTA improves planner performance and score map distribution by up to 30.0% and 8.5% respectively, particularly when CLIP predictions are poor due to limited training data and evaluation on *out-domain* taxonomies. We also demonstrate zero-shot generalization to text and sound modality without further fine-tuning the satellite image encoder with paired satellite image to text/sound data.

2 Related Works

Visual Navigation: There has been significant progress on visual navigation using purely vision [3–6] or vision-language foundation models (VLN) [7–9] to achieve high performance and generalization. VLNs are also used in the Object Navigation (*ObjectNav*) task, where a robot is required to search for objects of interest in indoor household environments. Before the emergence of VLNs, these search objects were limited to a closed set [10–12], but more recently, can extend to open sets described via natural language [32–35]. However, outdoor visual search, despite its relevance to tasks like path planning [36], exploration [37], or monitoring [38], remains relatively underexplored, with most prior works only working with closed-set targets [16–18]. While newer methods leverage foundation models to achieve better results [15, 39], they tend to have end-to-end architectures and require re-training when the vision backbone or planner changes. Instead, we focus on a modular approach to connect pre-trained VLMs to off-the-shelf search planners in a flexible manner.

Multimodal Learning: Ever since the emergence of powerful VLMs [20–22], there has been significant progress in training language foundation models with different modalities, such as audio [40–42], point clouds [43, 44], and to output action commands [45, 46]. In the remote sensing community, there has been significant interest in training language models with satellite images for semantic segmentation [47, 48], visual question-answer [49, 50] and predictive environmental monitoring [51, 52]. However, collecting aligned data across multiple modalities remains costly. Instead of training a single model, some works focus on chaining multiple models together to achieve multi-modality [53, 54], but may experience domain mismatch due to their pretraining on different datasets. Recent efforts align different modalities to a shared representation space [27–29] to achieve zero-shot generalization between modalities not jointly present in the training set. Our work explores this concept to achieve efficient visual search when prompted by inputs of varying modalities.

Online Adaptation: Online, or test-time adaptation (TTA), is essential for foundation models facing out-of-domain distributions, and has roots in prior works on continual learning [55–57]. In robotics, online adaptation is associated with meta-learning for few-shot learning [58, 59] and online adaptation to disturbances in robot dynamics [60]. Online adaptation is also related to replanning via Chain-of-Thought prompting [61] applied to text [62, 63] or vision [64, 65], to generate intermediate step-by-step explanations before providing better text/action output. Other approaches involve direct backpropagation to modify prompts [66–68], model weights [69, 70], and to handle dynamic distribution shifts [26, 71]. However, online adaptation with foundation models on satellite images is relatively underexplored. One example [72] uses a robot to navigate the scene using LLM-based traversability estimates, and uses feedback to update its prompts during navigation. Conversely, our work explores TTA for visual search using detection measurements to perform weight updates.

3 Problem Formulation

Environment: We adopt an outdoor variant of the *ObjectNav* formulation [10–12], where a robot is tasked with searching for multiple targets over a given satellite image within a time budget. First, we consider the search domain over a satellite map $\mathcal{S}$ as a grid map $\mathcal{M}$ composed of $n \times n$ uniform cells, where $\mathcal{M} = \{\psi_1, \psi_2, ...\}$ and ψ represents potential detection viewpoints corresponding to each cell on the map. We model the target distribution as a subset of grids $\mathcal{M}_t$, where $\mathcal{M}_t \subset \mathcal{M}$. Each grid may contain one or more targets, and these target locations are unknown to the robot *a-priori*.

Visual Priors: We generalize this formulation to accept search queries Q of different modalities, such as ground image Q_i, text Q_t, or sound Q_s. Pairs of input modalities $(\mathcal{S}, Q)$ are passed into a vision model to generate visual priors $p(T \mid \mathcal{S}, Q)$ to inform the search process. Such visual priors can take the form of embeddings in end-to-end frameworks [17], or predicted target probability distributions in frameworks where the vision and search modules are decoupled [16].

Target Search: Here, the robot is tasked to utilize the visual priors $p(T \mid \mathcal{S}, Q)$ to sequentially explore these cells $\mathcal{M}$ in order to determine the target locations $\mathcal{M}_t$. We model our target detection sensor to cover only the grid cell where the robot is currently located ψ_r. We define the trajectory of viewpoints for the robot $\psi = (\psi_1, \psi_2, \dots, \psi_m)$, $\psi_i \in \mathcal{M}$. This setup presents an optimization problem where we seek an optimal trajectory ψ^* given all possible trajectories Ψ, which maximizes the number of targets found given the budget constraint $\mathcal{B}$ number of steps. We denote the distance traveled by the robot as trajectory length $L(\psi)$, and utility U as the number of targets found.

$$\psi^* = \max_{\psi \in \Psi}(U), \quad \text{s.t. } L(\psi) \leq \mathcal{B} \tag{1}$$

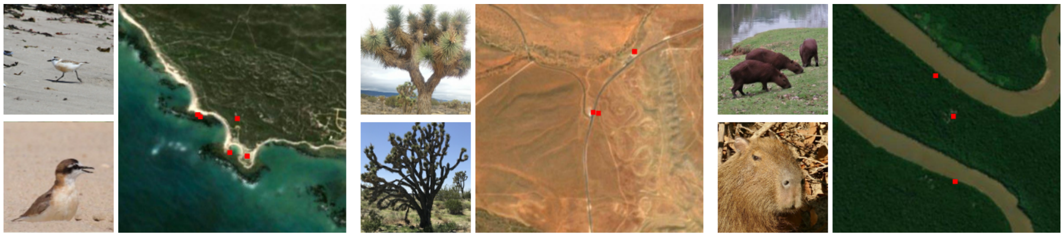


Figure 2: **Visual search dataset** The taxonomic targets cannot be directly seen on these satellite image [73], thus prompting the need to rely on visual cues to achieve efficient search [31].

4 AVS-Bench Ecological Dataset

There are limited datasets of satellite images with annotated targets that are not directly visible. To address this gap, we curate **AVS-Bench**, a visual search dataset based on internet-scale ecological data. It comprises Sentinel-2 level 2A satellite images [73] with unseen taxonomic targets from the iNat-2021 dataset [31], each tagged with ground-level image and taxonomic label (some with sound data). One advantage of using ecological data is the hierarchical structure of taxonomic labels (seven distinct tiers), which facilitates baseline evaluation across various levels of specificity. AVS-Bench is diverse in geography and taxonomies (Appendix A.1 & A.2) to reflect *in-the-wild* scenarios.

Taxonomic Location Dataset: Our goal is to generate a dataset where each image contains multiple target locations for the same taxonomy. We begin with the *iSatNat* dataset [29] with **2.7M** satellite images, each matching a ground-level image of a specific taxonomic label. We notice a significant amount of overlaps between satellite images, since taxonomies are located at the center of each image. Similar to [28], we apply a filter to obtain **441k** non-overlapping images and store the taxonomy-to-image mappings. Thereafter, we store a subset of images with ≥ 3 distinct landmarks (to focus on semantic-rich images) and within a range of 3-20 counts of the same taxonomy. We then split the remaining taxonomies from these filtered images equally into two distinct categories: *in-domain* and *out-domain* taxonomies. Using these new taxonomy categories, we further split the images into the **80k** training, **4k** *in-domain* validation, and **4k** *out-domain* validation datasets.

Taxonomic Score Maps: Existing VLMs are trained on large-scale datasets of natural images taken from egocentric viewpoints, and require fine-tuning to perform well on satellite images. However, there are limited remote sensing datasets that correlate segmentation masks with the likelihood of targets. Although it would be ideal to pretrain our VLMs using the taxonomic location dataset, they only include point locations, and conversion to segmentation masks with likelihood scores is non-trivial. We detail our procedures to convert the **80k** training dataset into score maps in Appendix A.3.

Training Dataset Usage: We finally train Search-TTA on **380k** *in-domain* images, obtained from the original **441k** non-overlapping images after excluding images from the validation sets and keeping only *in-domain* taxonomies. For our VLM baselines, we train them using our **80k** score maps.

5 Search-TTA Framework

We introduce **Search-TTA**, a multimodal test-time adaptation framework for AVS (Fig. 3), capable of generating and updating probability distributions while collecting measurements during the search process. We provide visualization and the algorithmic flow in Appendix B.1 and B.2 respectively.

5.1 Multimodal Score Map Generation

To accept queries of different modalities (e.g. text, image), we need to align their encoder outputs to the same representation space. We select the BioCLIP (ViT-B/16) [74] embedding space that is pretrained on the large-scale TreeOfLife dataset [74], which aligns taxonomy names to ground-level images. In addition, we train a satellite image encoder by fine-tuning a CLIP (ViT-L/14@336px) [20] image encoder to align with BioCLIP's embedding space. We achieve this alignment via the patch-level contrastive loss objective introduced in [75], which is a modified version of the standard In-

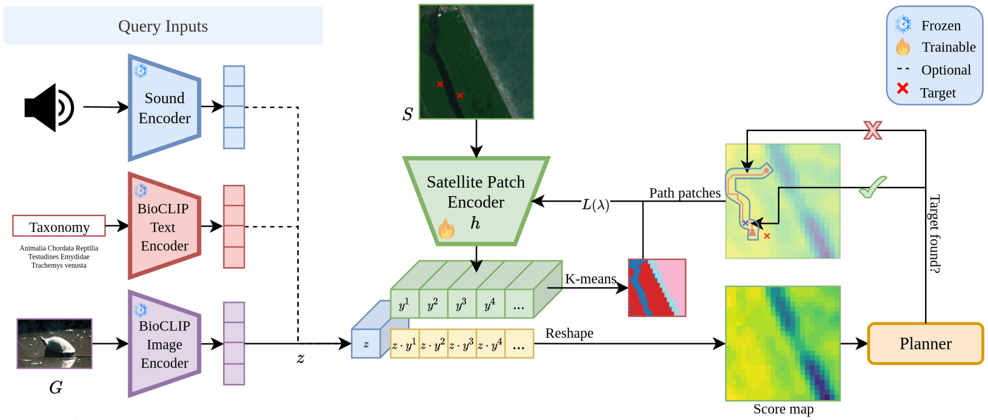


Figure 3: **Search-TTA Framework.** Given a satellite image, ground-level image, and inputs of other modalities, a score map is generated to guide the planner towards areas of high probability. The score map (initially poor) will be updated with SPPP gradient updates to the satellite patch encoder model during search.

foNCE loss [76] that performs alignment at image level. Intuitively, this objective aligns the features of the ground images closer with the features of their corresponding patches on the satellite map.

Consider the pair of modalities ($\mathcal{G}, \mathcal{S}$), where $\mathcal{G}$ denotes the ground-level image and $\mathcal{S}$ denotes the satellite image. Given a dataset containing numerous pairs of the two modalities, we organize them into mini-batches $\{g_i, s_i\}_{i=1,\ldots N}$ for training, where g and s refer to the ground images and their corresponding satellite patches respectively. We pass the inputs into their respective ground image encoder f and satellite image encoder h , and obtain their normalized embeddings as $z_i = f(g_i)$ and $y_i = h(s_i)$. Similar to [28], we remove the pooling layer prior to the final projection layer of the CLIP model to output a per-patch feature vector $z_i[p]$ across the entire satellite image. We then project $z_i[p]$ from the hidden dimension of 1024 to 512 to match the projection dimension of y_i. Subsequently, we compute the patch-level contrastive loss as such:

$$\mathcal{L}_{\mathcal{G}\to\mathcal{S}} = \frac{1}{N_B} \sum_{i \in N_B} \frac{1}{|G(i)|} \sum_{j \in G(i)} -\log \frac{\exp(z_i[p] \cdot y_j / \tau)}{\sum_{n \in N_B} \exp(z_i[p] \cdot y_n / \tau)} \tag{2}$$

where N_B is the total number of pairs in the mini-batch, $G(i)$ the set of ground images with the same species category as that of the ith satellite image, and τ the temperature parameter. Similarly, we define the patch-level contrastive loss $\mathcal{L}_{\mathcal{S}\to\mathcal{G}}$, and take the average of $\mathcal{L}_{\mathcal{G}\to\mathcal{S}}$ and $\mathcal{L}_{\mathcal{S}\to\mathcal{G}}$ as the final loss. Using AVS-Bench detailed in Sec. 4, we fine-tune the CLIP model with two NVIDIA A6000 GPUs, which took 3.5 days before convergence. During training, we update the weights of the satellite image encoder while keeping BioCLIP frozen. During inference, we generate the 24 × 24 probability distribution by taking the cosine distance between the query ground image features with all satellite image patch features. Further training details can be found in Appendix B.3.

5.2 Search Planners

Search-TTA is designed to be adaptable to different types of search planners. This ranges from conventional methods such as Information Surfing (IS) [77] to Deep Reinforcement Learning (RL) [78] methods. In principle, the Search-TTA framework can be applied to other types of planners as long as they can reasonably utilize the probability map from the vision model to inform their search strategies. We detail the performance of Search-TTA with various search planners in Section 6.1.

5.3 Test-Time Adaptation Feedback Loop

One of the key features of Search-TTA is its ability to refine probability distribution outputs from the vision model based on collected measurements. This process can be broken down into two stages.

K-means Clustering of CLIP Embeddings: Before the start of each search episode, we perform k-means clustering of the per-patch satellite image features $z_i[p]$ to generate clusters of embeddings that are semantically similar [79]. These clusters correspond to regions used in the modified Spatial

Table 1: Evaluating TTA on different planners (CLIP Vision Model), on Out-domain taxonomies

Planner Type	$\mathcal{B} = 256$							$\mathcal{B} = 384$						
	Found (%) ↑			**RMSE (%) ↓**			**Steps ↓**	**Found (%) ↑**			**RMSE (%) ↓**			**Steps ↓**
	All	***Bot. 5%***	***Bot. 2%***	***First***	***Mid***	***Last***	**(First tgt)**	***All***	***Bot. 5%***	***Bot. 2%***	***First***	***Mid***	***Last***	**(First tgt)**
RL (TTA) [78]	60.8	31.7	**30.7**	54.2	53.2	**51.2**	86.5	79.6	58.9	**56.1**	54.2	52.7	**47.0**	103.8
RL (no TTA) [78]	58.5	23.1	**16.0**	54.2	54.2	**54.2**	88.2	77.1	44.8	**36.1**	54.2	54.2	**54.2**	107.5
IS (TTA) [77]	53.9	24.2	**22.2**	54.2	52.7	**51.3**	92.7	74.0	46.1	**40.6**	54.2	52.7	**47.2**	114.8
IS (no TTA) [77]	51.2	19.1	**12.9**	54.2	54.2	**54.2**	92.3	71.9	32.3	**23.8**	54.2	54.2	**54.2**	115.6
Lawnmower [85]	41.7	–	–	–	–	–	118.8	74.2	–	–	–	–	–	157.6

Poisson Point Process (SPPP) [30] loss function below. This avoids the need to rely on external segmentation tools [80, 81]. In this work, we choose the best k by taking the average of the silhouette score criterion [82] and the elbow criterion [83]. More details can be found in Appendix B.4.

SPPP-based Online Adaptation: During search, the robot uses sensors to detect targets and collects feedback to refine CLIP's probability predictions. We adapt the Negative Log-Likelihood loss function from inhomogeneous SPPP to perform gradient updates for CLIP. SPPP is a statistical model to describe the frequency of scattered points in space with state-dependent intensity functions λ. While SPPP uses absolute λ values, CLIP's likelihood outputs can be approximated as normalized λ values across all regions. We can thus adapt SPPP's update function and apply it to CLIP.

Note that the vanilla loss function [30] does not work as a test-time update mechanism, because it was designed to regress SPPPs over a large batch of available data during training. In our case, the robot begins with no prior knowledge of the targets' locations and has to perform detection along the search process. For scenarios with sparse targets, the modes of the CLIP probability distribution may quickly collapse because the robot will collect many negative measurements before finding a first target. To address this, we introduce an uncertainty-driven weighting scheme that acts as a regularizer to the loss function, where p and n are the positive and negative measurements collected.

$$L(\lambda) = \sum_{i=1}^{p} (\alpha_{\text{pos},i}) \log \lambda(x_i) - \sum_{j=1}^{n} (\alpha_{\text{neg},j}) \, \lambda(x_j) \, dx. \tag{3}$$

Intuitively, we do not want to significantly reduce the probability of a semantic region after only a few negative detections, since they may not accurately represent the overall distribution. Hence, we scale negative measurements with the coefficient $\alpha_{\text{neg},j}$ based on how much of the corresponding semantic region has been covered. Similar to the concept of focal loss [84], we introduce an exponent γ (=2 in practice) to give less weight to measurements in regions that are largely uncovered. In practice, we use $\alpha_{\text{neg,j}} = \min\left(\beta \left(O_r / L_r\right)^{\gamma}, \, 1\right)$, where O_r is the number of patches observed in region r and L_r is the number of patches in that region. For $\alpha_{\text{pos},i}$, we find that keeping it constant (=4 in our case) works well in practice.

6 Experiments

The main objective of our experiments is to test Search-TTA's ability to enhance AVS performance, while providing a flexible plug-and-play interface compatible with various observation modalities and planning methods. We run our experiments using the AVS-Bench validation datasets.

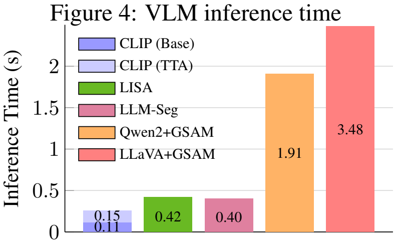
6.1 Effectiveness of TTA on Different Planners

We compare Search-TTA with an Attention-based RL planner [78] (pretrained on score maps from Sec. 4) and a greedy IS planner [77], while using Lawnmower [85] as a baseline, detailed in Appendix C.1. In Table 1, we report the average percentage of targets found, Root Mean Squared Error (RMSE) between CLIP predictions and ground truth score maps (from Sec. 4), and the steps taken to reach the first target, all within 256 steps and averaged across all **4k** *out-domain* validation images. To further determine Search-TTA's effectiveness, we recorded the targets found given poor CLIP predictions, namely in the bottom 5% and 2% percentiles in terms of CLIP prediction quality. To do so, we take the average scores of the pixels where the targets are located on the predicted score map and deem a CLIP prediction to be poor if most targets are located in low-scoring regions. Fig. C.1

Table 2: Comparing vision models (Found %) ↑

Vision Models	In-domain		Out-domain	
	$\mathcal{B} = 256$	$\mathcal{B} = 384$	$\mathcal{B} = 256$	$\mathcal{B} = 384$
CLIP (TTA) [74]	**57.4**	76.1	**60.8**	**79.6**
CLIP (no TTA) [74]	56.6	75.5	58.5	77.1
LISA [24]	57.1	**76.9**	58.4	77.8
LLM-Seg [86]	52.6	71.6	54.4	73.3
Qwen2+GroundedSAM [81, 87]	51.9	72.0	55.2	74.2
LLaVA+GroundedSAM [22, 81]	51.7	71.6	54.6	73.5

Figure 4: VLM inference time



and Fig. C.2 reflect that TTA performance gain is most significant in the bottom percentiles across all planners, indicating its ability to correct poor initial score maps *in-the-wild*.

Our results show a general improvement across all metrics with Search-TTA. We note the most significant improvement of 20.0% in the bottom 2% scoring CLIP predictions using the RL planner when $\mathcal{B} = 384$. Furthermore, we observe a decreasing trend in RMSE of up to 7.2%, which indicates that predicted score maps become increasingly accurate with TTA iterations. We note similar trends when evaluated on *in-domain* taxonomies (Table D.2), with RMSE improvements up to 8.5%.

6.2 Comparison with Baselines

Varying Vision Model: We evaluate the effectiveness of Search-TTA's CLIP vision backbone by replacing it with different state-of-the-art VLMs. We modify LISA and LLM-Seg to improve their performance for AVS, and fine-tune them with the score maps from Sec. 4. More details about these VLMs, training setup, and hyper-parameters can be found in Appendix C.2.

We run all of these baseline VLMs with our RL planner, and record the targets found (Table 2) and inference time (Fig. 4). Our results indicate that CLIP with TTA generally outperforms all baselines across different budgets except for *in-domain* data when $\mathcal{B} = 384$. In addition, we attribute LLM-Seg's poor performance to limitations in its training and output (training on binary masks only, and discretizing scores in its output maps). We also note that the fully decoupled baselines perform poorly, likely because Qwen-7B, LLaVA-13B, and GroundedSAM are not fine-tuned with remote sensing data. Note that CLIP has the fastest inference time (we run TTA only once every 20 steps).

AVS Baselines: We evaluate the effectiveness of Search-TTA by comparing it against existing AVS baselines (VAS and PSVAS) in the remote sensing domain. While VAS utilizes end-to-end reinforcement learning, PSVAS decouples vision and search models while introducing test-time adaptation (Appendix C.3).

Table 3: Comparing AVS frameworks (Found %)

Frameworks	Charadriiformes (In)		Columbiformes (Out)	
	$\mathcal{B} = 256$	$\mathcal{B} = 384$	$\mathcal{B} = 256$	$\mathcal{B} = 384$
CLIP+RL (TTA)	**60.3**	**79.7**	**62.9**	**82.2**
CLIP+RL (no TTA)	58.6	77.5	61.0	78.4
PSVAS [16]	53.0	68.5	60.3	75.0
VAS [17]	49.5	66.2	55.7	73.3
Lawnmower [85]	41.4	72.0	38.1	74.5

We evaluate our approach and the baselines using images with two different sub-classes of birds (*Animalia Chordata Aves*) as search targets, namely *Charadriiformes* and *Columbiformes*, which are more likely to be found along shorelines and on urban areas respectively. As seen in Table 3, CLIP with TTA and without TTA both significantly outperform the AVS baselines by up to 13.5% and 11.3% respectively. We note similar trends when evaluated on *Animalia Chordata Reptilia Squamata* (Appendix D.2). Although Lawnmower outperforms VAS and PSVAS when $\mathcal{B}$ =384, VAS and PSVAS can find the first target more quickly by performing a more targeted search (Table D.3).

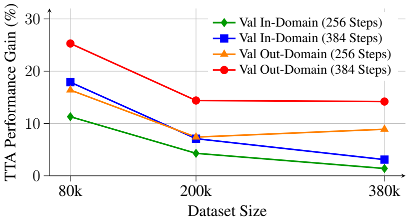
6.3 Multimodal Inputs

We evaluate the generalization ability of Search-TTA to previously unseen input modalities. To achieve this, we input the full taxonomic name into the CLIP text encoder, obtaining query text embeddings that are used in a manner similar to the ground image embeddings. We run these experiments over our *in-domain* validation images (Table 4), and note the performance gap of at most 0.9%. Separately, we fine-tune and evaluate a sound encoder [88] using the *quad-modal* split of AVS-Bench, achieving a performance gap of at most 2.4%. This indicates successful zero-shot generalization to text/sound modality although we did not fine-tune the satellite image encoder with text/sound data. More details on how we curated the sound dataset can be found in Appendix A.4.

Table 4: Zero-shot generalization (Found %)

Input Modality	Dataset Size	$\mathcal{B} = 256$		$\mathcal{B} = 384$	
		TTA	**No TTA**	**TTA**	**No TTA**
Image	4k	**57.4**	56.6	76.1	75.5
Text	4k	**56.7**	55.9	75.2	74.7
Image	460	56.2	55.0	75.1	**74.5**
Text	460	56.9	55.7	**76.5**	**74.7**
Sound	460	54.5	54.0	**75.1**	73.2

Figure 5: Dataset Scaling (Bot. 5%)



6.4 Ablation Studies

Scaling of Training Dataset: We analyze the impact of training dataset size for our satellite image encoder on TTA performance improvement. In Fig. 5, we observe that models trained on smaller datasets tend to benefit more from TTA. In Table C.2, we report up to a 30.0% increase in targets found with the 80k dataset, highlighting its strong correction abilities under limited training data.

SPPP Loss Coefficient: We investigate the effects of the vanilla log-likelihood loss function and hyperparameter tuning in Appendix B.5. When we remove the negative weighting coefficient ($\gamma = 0$) or the relative weighting factor ($\beta = 1$), we observe poorer performance of up to 8.1%.

Varying TTA methodology: We explore the effectiveness of our TTA methodology compared to prompt learning [68] and text-based TTA [64] in Appendix D.1. We notice that our SPPP-based formulation outperforms prompt learning in terms of targets found (Table C.3), and outperforms text-based TTA in terms of consistency in score map improvements during search (Table D.1).

6.5 Evaluation on Hardware

We carried out hardware-in-the-loop experiments to validate Search-TTA's performance in locating black bears in Yosemite Valley (Fig. 6). We deploy a Crazyflie 2.1 drone operating within a 4m × 4m mockup arena (17 × 17 grids) with external localization. Concurrently, we launch a ROS2 drone simulator [89] within a Yosemite Valley 3D model [90] in Gazebo, from which we obtain onboard measurements from the simulated drone's downward-facing camera, and use YOLO11x [91] for bear detection.

We conducted one experiment each for scenarios with and without TTA ($\mathcal{B} = 300s$), using a NAIP [1] satellite image of the operating area and an image of a bear sighting from iNaturalist [2] as inputs. From Fig. 1, we note that 5 targets were found with TTA, compared to 3 targets found without TTA. With TTA, the detection of the first bear significantly corrected this initial distribution, guiding the robot to explore the dense forested areas and thus find more targets. More experimental details can be found in Appendix D.2.

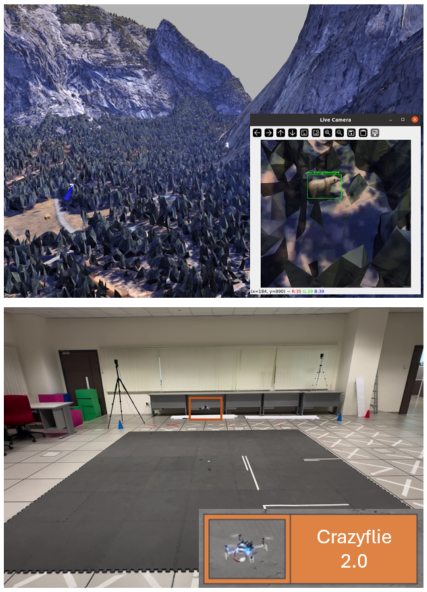


Figure 6: **AVS with Crazyflie drone (perception simulated in Gazebo).**

7 Conclusion

This paper addresses the challenges of autonomous outdoor visual search where targets cannot be seen directly from satellite images. We introduce **Search-TTA**, a test-time adaptation framework that enhances potentially inaccurate CLIP predictions. Our contributions include curating **AVS-Bench**, an internet-scale dataset with unseen taxonomic targets, enabling multimodal query through alignment to a common representation space, and proposing a novel TTA mechanism inspired by Spatial Poisson Point Processes. Search-TTA significantly improves planner performance and score map distribution by up to 30.0% and 8.5% respectively. We also demonstrate zero-shot generalization to text and sound modality without additional fine-tuning. We hope that our research will inspire future work in visual search and ecological conservation.

Acknowledgments

This work was supported by Singapore Technologies Engineering Ltd, under the Economic Development Board - Industrial Postgraduate Program (Project No. 2022-2130). We would like to extend our gratitude to Kimberly Fornace (School of Public Health) and Adrian Loo Hock Beng (Faculty of Science, Center for Nature-based Climate Solutions) from the National University of Singapore for the helpful conversations on ecological conservation that shaped the early stages of this paper.

8 Limitations and Future Work

Experimental Realism: There are a very limited number of existing works dealing with AVS over satellite images. In order to fairly compare with [16, 17], we decided to remain consistent with their problem statement, and naturally inherit some of the limitations in their formulation:

- We assume that our sensor model has very narrow field of view. As a result, search performance can be highly stochastic, making performance improvements quite marginal (especially when averaged over a large validation set), as robots can easily miss targets by a small margin.
- We assume that our sensor model is perfect and binary. This does not account for detection uncertainty or false positive/negative measurements, which would require robots to re-visit specific areas to confirm their beliefs. Future work will look at extending our TTA method to handle more complex and realistic search formulations.
- We assume that our sensor model is capable of detecting targets which may be hidden in areas occluded from direct satellite imagery (e.g. dense forest, water surfaces etc.). Future work will consider other sensor modalities (e.g. thermal camera, camera traps etc.) or other types of robots (e.g. unmanned ground/underwater vehicles) to model realistic detection constraints.

Beyond Visual Semantics: Search-TTA is effective in drawing connections between target taxonomy and visual semantics, but does not currently consider other essential factors when determining the likelihood of their whereabouts, such as relationships between the different landmarks, interactions between different species, sources of food/danger, etc. Future work will extend the reasoning of our VLM to allow for such deeper, multi-faceted reasoning.

Multi-Target Search: Adapting Search-TTA to simultaneously search for multiple target types presents significant challenges related to catastrophic forgetting [92] during gradient updates when different target types are encountered sequentially. Future work will focus on developing continual learning methodologies (e.g. batch normalization or importance sampling [26]) that enable our framework to maintain performance across previously learned targets while adapting to new ones.

References

[1] U.S.G.S. National agriculture imagery program (naip), 2022.

[2] iNaturalist. inaturalist. URL `https://www.inaturalist.org`.

[3] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine. ViNT: A foundation model for visual navigation. In *7th Annual Conference on Robot Learning*, 2023.

[4] A. Sridhar, D. Shah, C. Glossop, and S. Levine. NoMaD: Goal Masked Diffusion Policies for Navigation and Exploration. *arXiv pre-print*, 2023.

[5] A. Bar, G. Zhou, D. Tran, T. Darrell, and Y. LeCun. Navigation world models, 2024.

[6] D. Shah and S. Levine. ViKiNG: Vision-Based Kilometer-Scale Navigation with Geographic Hints. In *Proceedings of Robotics: Science and Systems*, 2022.

[7] J. Gu, E. Stefani, Q. Wu, J. Thomason, and X. Wang. Vision-and-language navigation: A survey of tasks, methods, and future directions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7606–7623, 2022. doi:10.18653/v1/2022.acl-long.524.

[8] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2018. doi:10.1109/CVPR.2018.00387.

[9] D. Shah, B. Osinski, B. Ichter, and S. Levine. LM-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *6th Annual Conference on Robot Learning*, 2022.

[10] Z. Zeng, A. Röfer, and O. C. Jenkins. Semantic linking maps for active visual object search. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1984–1990, 2020. doi:10.1109/ICRA40945.2020.9196830.

[11] D. S. Chaplot, D. Gandhi, A. Gupta, and R. Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *In Neural Information Processing Systems*, 2020.

[12] A. J. Zhai and S. Wang. PEANUT: Predicting and navigating to unseen targets. In *ICCV*, 2023.

[13] K. Koreitem, F. Shkurti, T. Manderson, W.-D. Chang, J. C. G. Higuera, and G. Dudek. One-shot informed robotic visual search in the wild. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5800–5807. IEEE, 2020.

[14] F. Niroui, K. Zhang, Z. Kashino, and G. Nejat. Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments. *IEEE Robotics and Automation Letters*, 4(2):610–617, 2019.

[15] H. Wang, A. H. Tan, and G. Nejat. Navformer: A transformer architecture for robot target-driven navigation in unknown and dynamic environments. *IEEE Robotics and Automation Letters*, 9(8):6808–6815, 2024. doi:10.1109/LRA.2024.3412638.

[16] A. Sarkar, N. Jacobs, and Y. Vorobeychik. A partially-supervised reinforcement learning framework for visual active search. *Advances in Neural Information Processing Systems*, 36: 12245–12270, 2023.

[17] A. Sarkar, M. Lanier, S. Alfeld, J. Feng, R. Garnett, N. Jacobs, and Y. Vorobeychik. A visual active search framework for geospatial exploration. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 8316–8325, 2024.

[18] Y. Wang, Y. Wang, Y. Cao, and G. Sartoretti. Spatio-temporal attention network for persistent monitoring of multiple mobile targets. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3903–3910, 2023. doi:10.1109/IROS55552.2023.10341674.

[19] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[20] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[21] J. Li, D. Li, S. Savarese, and S. Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023.

[22] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[23] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[24] X. Lai, Z. Tian, Y. Chen, Y. Li, Y. Yuan, S. Liu, and J. Jia. Lisa: Reasoning segmentation via large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9579–9589, 2024.

[25] V. Rawte, A. Sheth, and A. Das. A survey of hallucination in ”large” foundation models, 2023.

[26] L. Yuan, B. Xie, and S. Li. Robust test-time adaptation in dynamic scenarios. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15922–15932, 2023.

[27] R. Girdhar, A. El-Nouby, Z. Liu, M. Singh, K. V. Alwala, A. Joulin, and I. Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15180–15190, 2023.

[28] U. Mall, C. P. Phoo, M. K. Liu, C. Vondrick, B. Hariharan, and K. Bala. Remote sensing vision-language foundation models without annotations via ground remote alignment. *arXiv preprint arXiv:2312.06960*, 2023.

[29] S. Sastry, S. Khanal, A. Dhakal, A. Ahmad, and N. Jacobs. Taxabind: A unified embedding space for ecological applications. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1765–1774. IEEE, 2025.

[30] P. Diggle. *Statistical analysis of spatial and spatio-temporal point patterns, third edition*. 07 2013. ISBN 9780429098093. doi:10.1201/b15326.

[31] G. Van Horn and O. Mac Aodha. inat challenge 2021 - fgvc8, 2021. URL `https://kaggle.com/competitions/inaturalist-2021`.

[32] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23171–23181, 2023. doi:10.1109/CVPR52729.2023.02219.

[33] B. Yu, H. Kasaei, and M. Cao. L3mvn: Leveraging large language models for visual target navigation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3554–3560, 2023. doi:10.1109/IROS55552.2023.10342512.

[34] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 42–48, 2024. doi:10.1109/ICRA57147.2024.10610712.

[35] M. Chang, T. Gervet, M. Khanna, S. Yenamandra, D. Shah, S. Y. Min, K. Shah, C. Paxton, S. Gupta, D. Batra, R. Mottaghi, J. Malik, and D. Chaplot. Goat: Go to any thing. 2024. doi:10.15607/RSS.2024.XX.073.

[36] S. Manjanna, J. Hansen, A. Q. Li, I. Rekleitis, and G. Dudek. Collaborative sampling using heterogeneous marine robots driven by visual cues. In *2017 14th Conference on Computer and Robot Vision (CRV)*, 2017.

[37] D. M. S. Tan, Y. Ma, J. Liang, Y. C. Chng, Y. Cao, and G. Sartoretti. Ir 2: Implicit rendezvous for robotic exploration teams under sparse intermittent connectivity. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13245–13252. IEEE, 2024.

[38] J. Chiun, S. Zhang, Y. Wang, Y. Cao, and G. Sartoretti. Marvel: Multi-agent reinforcement learning for constrained field-of-view multi-robot exploration in large-scale environments. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[39] A. Sarkar, S. Sastry, A. Pirinen, C. Zhang, N. Jacobs, and Y. Vorobeychik. Gomaa-geo: Goal modality agnostic active geo-localization. *arXiv preprint arXiv:2406.01917*, 2024.

[40] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever. Robust speech recognition via large-scale weak supervision, 2022.

[41] A. Guzhov, F. Raue, J. Hees, and A. Dengel. Audioclip: Extending clip to image, text and audio. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 976–980, 2022. doi:10.1109/ICASSP43922.2022.9747631.

[42] V. Pratap, A. Tjandra, B. Shi, P. Tomasello, A. Babu, S. Kundu, A. Elkahky, Z. Ni, A. Vyas, M. Fazel-Zarandi, A. Baevski, Y. Adi, X. Zhang, W.-N. Hsu, A. Conneau, and M. Auli. Scaling speech technology to 1,000+ languages. *J. Mach. Learn. Res.*, 25(1), 2024.

[43] R. Xu, X. Wang, T. Wang, Y. Chen, J. Pang, and D. Lin. Pointllm: Empowering large language models to understand point clouds. In *ECCV*, 2024.

[44] S. Yang, J. Liu, R. Zhang, M. Pan, Z. Guo, X. Li, Z. Chen, P. Gao, H. Li, Y. Guo, and S. Zhang. Lidar-llm: Exploring the potential of large language models for 3d lidar understanding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39:9247–9255, 2025. doi:10.1609/aaai.v39i9.33001.

[45] M. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[46] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024.

[47] C. Ye, Y. Zhuge, and P. Zhang. Towards open-vocabulary remote sensing image semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

[48] F. Trujillano, G. Jiménez, E. J. Manrique Valverde, N. Kahamba, F. Okumu, N. Apollinaire, G. Carrasco-Escobar, B. Barrett, and K. Fornace. Using image segmentation models to analyse high-resolution earth observation data: new tools to monitor disease risks in changing environments. *International Journal of Health Geographics*, 23, 05 2024. doi:10.1186/s12942-024-00371-w.

[49] K. Kuckreja, M. S. Danish, M. Naseer, A. Das, S. Khan, and F. S. Khan. Geochat: Grounded large vision-language model for remote sensing. *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[50] A. Dhakal, A. Ahmad, S. Khanal, S. Sastry, H. Kerner, and N. Jacobs. Sat2cap: Mapping fine-grained textual descriptions from satellite images. In *IEEE/ISPRS Workshop: Large Scale Computer Vision for Remote Sensing (EARTHVISION)*, pages 533–542, 2024.

[51] C. Brown, M. Kazmierski, V. Pasquarella, W. Rucklidge, M. Samsikova, C. Zhang, E. Shelhamer, E. Lahera, O. Wiles, S. Ilyushchenko, N. Gorelick, L. Zhang, S. Alj, E. Schechter, S. Askay, O. Guinan, R. Moore, A. Boukouvalas, and P. Kohli. Alphaearth foundations: An embedding field model for accurate and efficient global mapping from sparse label data. 2025. doi:10.48550/arXiv.2507.22291.

[52] C. Bodnar, W. P. Bruinsma, A. Lucic, M. Stanley, A. Allen, J. Brandstetter, P. Garvan, M. Riechert, J. A. Weyn, H. Dong, J. K. Gupta, K. Thambiratnam, A. T. Archibald, C.-C. Wu, E. Heider, M. Welling, R. E. Turner, and P. Perdikaris. A foundation model for the earth system. *Nature*, 2025. doi:10.1038/s41586-025-09005-y.

[53] A. Zeng, M. Attarian, brian ichter, K. M. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. S. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, and P. Florence. Socratic models: Composing zero-shot multimodal reasoning with language. In *The Eleventh International Conference on Learning Representations*, 2023.

[54] Z. Yang, L. Li, J. Wang, K. Lin, E. Azarnasab, F. Ahmed, Z. Liu, C. Liu, M. Zeng, and L. Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action. 2023.

[55] M. Woł czyk, M. Zajac, R. Pascanu, L. u. Kuciński, and P. Mił oś. Continual world: A robotic benchmark for continual reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 28496–28510, 2021.

[56] Y. Huang, K. Xie, H. Bharadhwaj, and F. Shkurti. Continual model-based reinforcement learning with hypernetworks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 799–805, 2021. doi:10.1109/ICRA48506.2021.9560793.

[57] Y. Meng, Z. Bing, X. Yao, K. Chen, K. Huang, Y. Gao, F. Sun, and A. Knoll. Preserving and combining knowledge in robotic lifelong reinforcement learning. *Nature Machine Intelligence*, pages 1–14, 2025.

[58] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, page 1126–1135, 2017.

[59] A. Zhou, E. Jang, D. Kappler, A. Herzog, M. Khansari, P. Wohlhart, Y. Bai, M. Kalakrishnan, S. Levine, and C. Finn. Watch, try, learn: Meta-learning from demonstrations and rewards. In *International Conference on Learning Representations*, 2020.

[60] I. Clavera, A. Nagabandi, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019.

[61] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2022. ISBN 9781713871088.

[62] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *arXiv preprint arXiv:2307.15818*, 2023.

[63] Y. Mu, Q. Zhang, M. Hu, W. Wang, M. Ding, J. Jin, B. Wang, J. Dai, Y. Qiao, and P. Luo. Embodiedgpt: vision-language pre-training via embodied chain of thought. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, 2023.

[64] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, N. Brown, T. Jackson, L. Luu, S. Levine, K. Hausman, and B. Ichter. Inner monologue: Embodied reasoning through planning with language models. In *arXiv preprint arXiv:2207.05608*, 2022.

[65] M. Skreta, Z. Zhou, J. L. Yuan, K. Darvish, A. Aspuru-Guzik, and A. Garg. Replan: Robotic replanning with perception and language models, 2024.

[66] B. Lester, R. Al-Rfou, and N. Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[67] X. Liu, K. Ji, Y. Fu, W. L. Tam, Z. Du, Z. Yang, and J. Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

[68] M. Shu, W. Nie, D.-A. Huang, Z. Yu, T. Goldstein, A. Anandkumar, and C. Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. *Advances in Neural Information Processing Systems*, 35:14274–14289, 2022.

[69] S. Zhao, X. Wang, L. Zhu, and Y. Yang. Test-time adaptation with clip reward for zero-shot generalization in vision-language models. *arXiv preprint arXiv:2305.18010*, 2023.

[70] J. Song, J. Lee, I. S. Kweon, and S. Choi. Ecotta: Memory-efficient continual test-time adaptation via self-distilled regularization. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11920–11929, 2023. doi:10.1109/CVPR52729.2023.01147.

[71] S. Niu, J. Wu, Y. Zhang, Z. Wen, Y. Chen, P. Zhao, and M. Tan. Towards stable test-time adaptation in dynamic wild world. In *Internetional Conference on Learning Representations*, 2023.

[72] M. Elnoor, K. Weerakoon, G. Seneviratne, R. Xian, T. Guan, M. K. M. Jaffar, V. Rajagopal, and D. Manocha. Robot navigation using physically grounded vision-language models in outdoor environments, 2024.

[73] H. Van der Werff and F. Van der Meer. Sentinel-2a msi and landsat 8 oli provide data continuity for geological remote sensing. *Remote sensing*, 8(11):883, 2016.

[74] S. Stevens, J. Wu, M. J. Thompson, E. G. Campolongo, C. H. Song, D. E. Carlyn, L. Dong, W. M. Dahdul, C. Stewart, T. Berger-Wolf, et al. Bioclip: A vision foundation model for the tree of life. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19412–19424, 2024.

[75] S. Zhang, F. Zhu, R. Zhao, and J. Yan. Patch-level contrasting without patch correspondence for accurate and dense contrastive representation learning. *arXiv preprint arXiv:2306.13337*, 2023.

[76] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[77] P. Lanillos, S. K. Gan, E. Besada-Portas, G. Pajares, and S. Sukkarieh. Multi-uav target search using decentralized gradient-based negotiation with expected observation. *Information Sciences*, 282:92–110, 2014.

[78] Y. Cao, T. Hou, Y. Wang, et al. Ariadne: A reinforcement learning approach using attention-based deep networks for exploration. In *2023 IEEE ICRA*, 2023.

[79] J. Ma, P.-Y. Huang, S. Xie, S.-W. Li, L. Zettlemoyer, S.-F. Chang, W.-T. Yih, and H. Xu. Mode: Clip data experts via clustering. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[80] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

[81] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.

[82] K. R. Shahapure and C. Nicholas. Cluster quality analysis using silhouette score. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 747–748, 2020.

[83] D. Marutho, S. Hendra Handaka, E. Wijaya, and Muljono. The determination of cluster number at k-mean using elbow method and purity evaluation on headline news. In *2018 International Seminar on Application for Technology of Information and Communication*, pages 533–538, 2018.

[84] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection, 2018.

[85] H. Choset. Coverage for robotics - a survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31:113 – 126, October 2001.

[86] J. Wang and L. Ke. Llm-seg: Bridging image segmentation and large language model reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1765–1774, 2024.

[87] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[88] Y. Wu, K. Chen, T. Zhang, Y. Hui, M. Nezhurina, T. Berg-Kirkpatrick, and S. Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. 2024.

[89] H. Huang and J. Sturm. tum_simulator. URL `https://wiki.ros.org/tum_simulator`.

[90] Google. Google earth engine.

[91] G. Jocher and J. Qiu. Ultralytics yolo11, 2024. URL `https://github.com/ultralytics/ultralytics`.

[92] W. Hu, Z. Lin, B. Liu, C. Tao, Z. Tao, J. Ma, D. Zhao, and R. Yan. Overcoming catastrophic forgetting via model adaptation. In *International Conference on Learning Representations*, 2019.

[93] A. Garioud, S. Peillet, E. Bookjans, S. Giordano, and B. Wattrelos. Flair #1: semantic segmentation and domain adaptation dataset. 2022. doi:10.13140/RG.2.2.30183.73128/1.

[94] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: Training imagenet in 1 hour, 2018.

[95] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine. Soft actor-critic algorithms and applications, 2019.

[96] D. M. S. Tan, A. Rao, A. Breitfeld, and G. Sartoretti. Context mask priors via vision-language model for ergodic search. In *IEEE International Conference on Robotics and Automation (Workshop on Ergodic Control)*, 2024. URL `https://github.com/search-tta/context-mask-search-priors`.

[97] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[98] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models, 2020.

[99] E. J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Appendix: Supplementary Material

A AVS-Bench Dataset Details

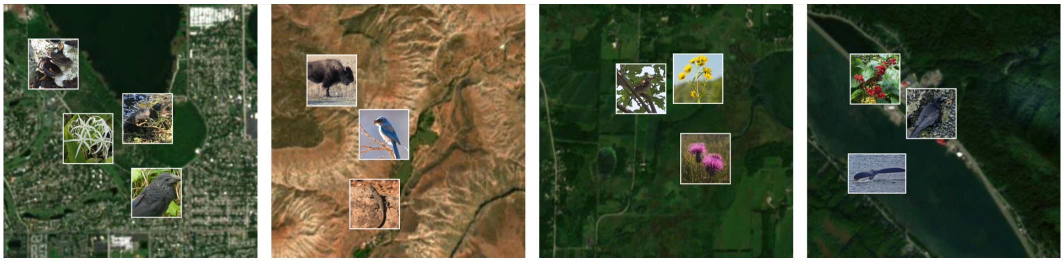


Figure A.1: Examples of satellite images [73] in the full **380k** dataset (each with different taxonomies [31]), used for CLIP fine-tuning.

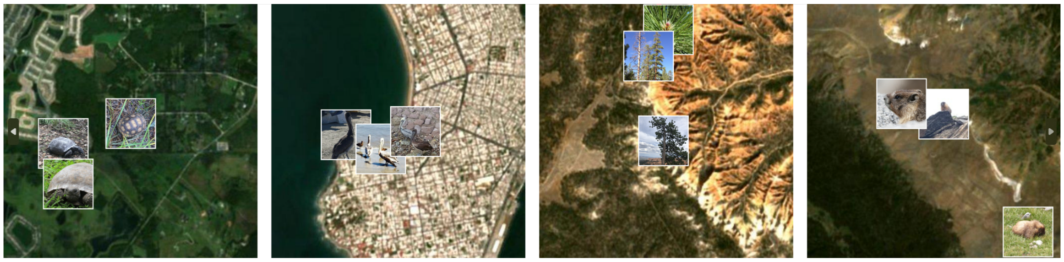


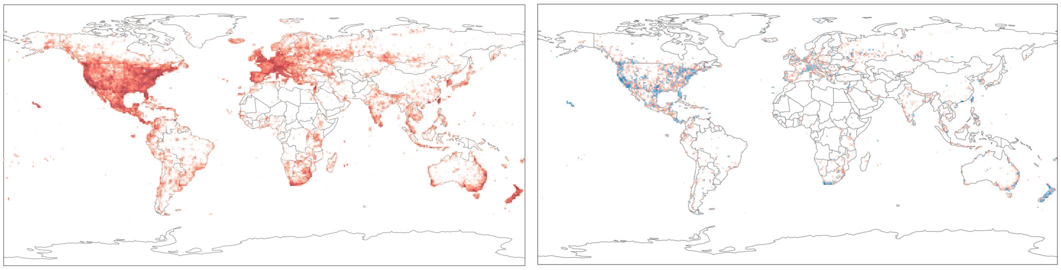
Figure A.2: Examples of satellite images [73] in the **80k** training and **4k** validation datasets (each with same taxonomies [31]), used for AVS validation and score maps generation.

In this section, we provide more details about our AVS-Bench dataset composition and generation process, in addition to the information provided in Sec. 4. Our *tri-modal* dataset split contains Sentinel-2 level 2A satellite images [73] covers approximately 2.56km×2.56km of land mass, each with targets that are paired with their taxonomic names, locations, and ground images.

1. **CLIP training dataset**: 380k satellite images with different taxonomic targets (Fig. A.1).
2. **AVS training dataset**: 80k satellite images with same taxonomic targets (Fig. A.2).
3. **AVS validation datasets**: 4k satellite images with same taxonomic targets that are *in-domain*, and 4k satellite images with same taxonomic targets that are *out-domain* (Fig. A.2).

A.1 Geographical Coverage

We visualize the spatial distribution of our dataset in Fig. A.3, where the color intensity reflects the taxonomy counts in each cell (1° latitude $\times$ 1° longitude). Despite filtering our dataset to cater to our AVS task, the **80k** training and **4k** *in-domain* validation datasets appear visually representative of the original **380k** dataset distribution (**4k** *out-domain* validation dataset has a similar distribution).



380k full training dataset

80k train (Red) & **4k** validation (Blue) datasets

Figure A.3: Geographic coverage of datasets used in training and validation.

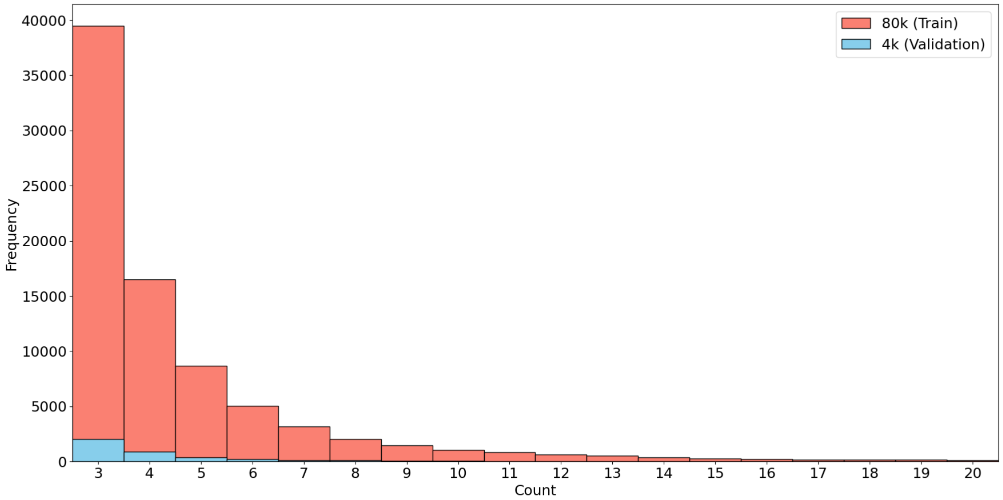


Figure A.4: Histogram counts for **80k** train (Red) & **4k** *out-domain* validation (Blue) datasets

Table A.1: Taxonomy distribution across training and validation datasets

Taxonomies	**Train (380k)**		**Train (80k)**		**Val In-Domain (4k)**		**Val Out-Domain (4k)**	
	Targets (%)	Images (%)	Targets (%)	Images (%)	Targets (%)	Images (%)	Targets (%)	Images (%)
Plants	43.8	–	45.3	45.8	51.1	50.9	39.7	40.1
Insects	30.1	–	29.8	29.5	16.9	17.1	1.1	1.2
Birds	12.7	–	11.5	11.5	17.4	17.3	18.7	19.8
Fungi	3.5	–	2.7	2.9	2.5	2.6	1.9	2.1
Reptiles	3.8	–	3.5	3.4	4.2	4.2	0.7	0.6
Mammals	0.8	–	1.1	1.1	0.9	1.0	8.4	8.3
Fishes	1.0	–	1.7	1.5	1.8	1.9	3.7	3.7
Amphibians	2.1	–	2.4	2.3	1.8	1.9	0.0	0.0
Mollusks	0.2	–	0.5	0.4	0.6	0.6	14.9	14.2
Arachnids	1.6	–	1.0	1.1	1.1	1.1	0.5	0.5
Animalia	0.4	–	0.5	0.5	1.6	1.7	10.3	9.6
Total	2,601,787	379,962	365,292	80,535	13,334	4000	17,188	4000

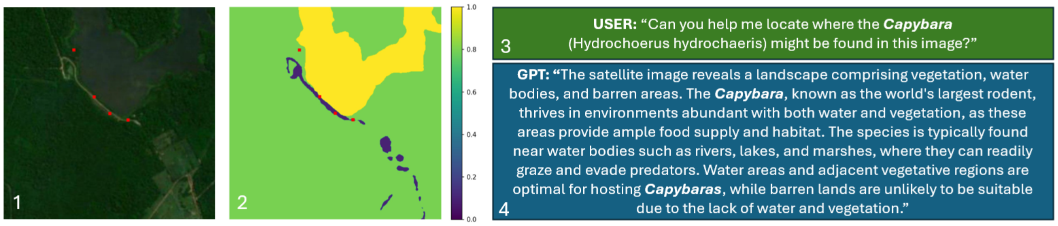
A.2 Taxonomy Statistics

We visualize the distribution of the taxonomy counts for the training and validation datasets in Fig. A.4. We note a natural decreasing trend in the number of same-taxonomy targets within the 2.56km×2.56km of land mass covered by each image. The average number of taxonomic counts per image for our **80k** training datset is 4.5±2.6, **4k** *in-domain* validation dataset is 3.3±0.7, and **4k** *out-domain* validation dataset is 4.3±2.3. This results in a very sparse target distribution per image, making our AVS task challenging given a budget constraint.

In addition, we visualized the breakdown of the taxonomy categories for all of our datasets in Table A.1. These distributions, except for the *out-domain* validation dataset, are similar to the dataset distribution from the iNaturalist 2021 challenge where these datasets originate from [31].

A.3 Score Map Generation

We generate our **80k** training score masks using a custom process because our AVS dataset only includes point locations, and conversion to segmentation masks with likelihood scores is non-trivial. This is done in two stages. First, we use GSNet [47], an open-vocabulary semantic segmentation model, to obtain label maps of the low-resolution Sentinel-2 images based on broad landmark names (i.e. *Urban*, *Water*, *Vegetation*, *Barren*). Since GSNet has been pretrained on a diverse set of satellite images with varying spatial resolution, we fine-tune it with the FLAIR semantic segmentation dataset [93] to enhance its segmentation abilities specifically for low-resolution images. However, despite the high-quality segmentation masks, mislabeling commonly occurs. Hence, we



USER: "Can you help me locate where the ***Capybara*** (Hydrochoerus hydrochaeris) might be found in this image?"

GPT: "The satellite image reveals a landscape comprising vegetation, water bodies, and barren areas. The ***Capybara***, known as the world's largest rodent, thrives in environments abundant with both water and vegetation, as these areas provide ample food supply and habitat. The species is typically found near water bodies such as rivers, lakes, and marshes, where they can readily graze and evade predators. Water areas and adjacent vegetative regions are optimal for hosting ***Capybaras***, while barren lands are unlikely to be suitable due to the lack of water and vegetation."

Figure A.5: (1) Satellite map [73] where Capybaras can be found [31]. (2) Score map of where Capybaras are likely to be found, used for fine-tuning VLM baselines and training RL policy. (3-4) Question and answer pair, used to fine-tune VLMs such as LISA [24].

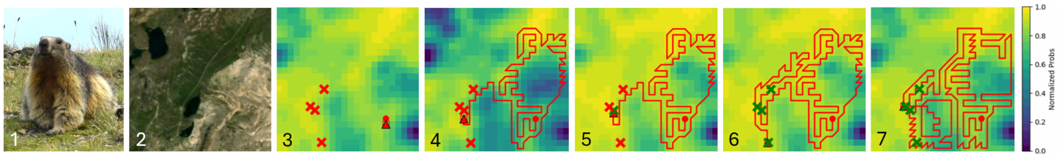


Figure A.6: (1) Ground image of a *Marmot*. (2) Satellite image where marmots can be found. (3) Initial probability output. (4) TTA enables regulated decrease in probability values in regions where marmots are not detected. (5) TTA significantly increases probability values in region where the first Marmot is found. (6) Improved priors leads to efficient search. (7) Inefficient search without TTA.

use GPT4o [23], augmented with human-labeled examples, to rectify the wrong labels. We then use GPT4o again to score the masks based on the likelihood of finding the specified taxonomies per image. It concurrently generates a conversation that explains the rationale for why certain landmarks are more suitable for the specified taxonomies, and less so for others. After filtering erroneous GPT4o generations, we end up with **80k** score maps for each taxonomic-image pair. An example can be seen in Fig. A.5 (prompts used for relabeling and scoring by GPT4o are shown in Appendix E.1).

A.4 Sound Dataset Generation

We further fine-tune a sound encoder in order to evaluate the generalization ability of Search-TTA to previously unseen input modalities (Sec. 6.3). We follow a similar process as in Sec. 4 to generate our *quad-modal* training and validation datasets. We begin with the iSoundNat dataset [29] with **74.9k** satellite images, each matching a ground-level image, a taxonomic label, and a sound recording. We keep only the *in-domain* taxonomies defined in Sec. 4, and end up with **68.8k** data points used for fine-tuning the CLAP sound encoder [88]. To validate the its performance, we curate a new validation dataset by keeping data from the **4k** in-domain validation datasets that contain targets with sound data. By the end of this process, we have an in-domain validation dataset with **480** data points containing all modalities including sound. Training details can be found in Appendix B.3.

B Additional Search-TTA Details

In this section, we provide more details of the Search-TTA framework, to supplement the information in Sec. 5. We elaborate on how Search-TTA generates and periodically updates the probability distribution outputs that serve as a strong prior to the RL search planner.

B.1 Qualitative Analysis

We provide snapshots of AVS with the RL planner to provide a better understanding of how it works. In Fig. A.6, the RL planner begins with an initial prior generated by the satellite image CLIP encoder, which represents the probability distribution of where *Marmots* (*Mammalia Rodentia Sciuridae Marmota*) can be found. Within the first phase, TTA enables a regulated decrease in probability values within the regions where marmots are not detected. Thereafter, Search-TTA significantly improves the probability distribution outputs in the associated region where the first and subsequent targets are found. This steers the RL planner to exploit the high probability region to locate all of the targets within 181 steps. Without TTA, the probability distribution is static and the RL planner takes 242 steps before locating all marmots.

B.2 Algorithm

We provide the pseudo-code to illustrate the Search-TTA framework (algorithm 1).

Algorithm 1: The Search-TTA Framework (with RL Search Policy)

```
Input: Satellite-image encoder $f_\theta$; Ground-image encoder $h_\psi$; Search policy $\pi_\phi$,
       Satellite image $s$, Ground image $g$, Budget $\mathcal{B}$
Initialize: $\theta \leftarrow \theta_{\text{base}}$, step $t = 0$, measurements $O = [0, \ldots, 0]$, grid map $\mathcal{M} = (n \times n)$, $\alpha_{\text{pos},i} = 4$
// --- Generate Probability Distribution ---
$z \leftarrow f_\theta(s)$, $y \leftarrow h_\psi(g)$;
$P \leftarrow \text{cosineSim}(z, y)$;
$r \leftarrow \text{kmeans}(z)$;
while $t \leq \mathcal{B}$ do
    // --- Generate Action ---
    $a_t \sim \pi_\phi(\cdot \mid o_t, P)$;
    $s_{t+1} \sim T(o_t, a_t)$;
    // --- Collect Measurements ---
    $d \leftarrow \text{Observe}(o_{t+1})$;
    $O \leftarrow \text{Update}(O, d)$;
    // --- Perform TTA ---
    for every $k$ steps do
        $\theta \leftarrow \theta_{\text{base}}$;
        $\gamma \leftarrow \gamma_{\min} + (t/n^2) \cdot (\gamma_{\max} - \gamma_{\min})$;
        $\alpha_{\text{neg},j} \leftarrow \min\left(\beta \left(O_r/L_r\right)^\gamma, 1\right)$;
        $L(\lambda) = \sum_{i=1}^{p} \alpha_{\text{pos},i} \cdot \log \lambda(x_i) - \sum_{j=1}^{n} \alpha_{\text{neg},j} \cdot \lambda(x_j)\, dx$, where $P \approx \lambda$;
        $\theta \leftarrow \theta + \gamma \cdot \nabla_\theta L(\lambda)$;
```

B.3 Training Details for Satellite Image and Sound Encoders

We fine-tune our satellite image CLIP encoder [20] with the hyperparameters in Table B.1. This was performed using two NVIDIA A6000 GPUs, which took 3 epochs (3.5 days) before convergence (lowest CLIP validation score). In addition, we fine-tune our CLAP sound encoder [88] with similar hyperparameters. This was performed using four NVIDIA A5000 GPUs, which took 19 epochs (11 hours). While fine-tuning both encoders, we keep our BioCLIP [74] model frozen.

Table B.1: Training hyperparameters (query encoder)

Hyperparameter	**Value**
Batch Size	32
Learning Rate	1e-4
Learning Rate Schedule	min 1e-6 (Cosine Annealing)
Temperature (τ)	0.07
Optimizer	AdamW
Optimizer β	(0.9, 0.98)
Optimizer ϵ	1e-6
Accumulate Grad batches	64
Projection Dimension	512
Ground Image Encoder	BioCLIP [74] (ViT-B/16)
Satellite Image Encoder	CLIP [20] (ViT-L/14@336px)
Sound Encoder	CLAP [88]

B.4 Kmeans Clustering

We rely on Kmeans clustering of the satellite image encoder output to obtain clusters of embeddings that are deemed semantically similar by CLIP [79]. We determine the best k by taking the average of the silhouette score criterion [82] and the elbow criterion [83]. The silhouette score measures clustering quality by contrasting each satellite patch feature's average distance to its

Table B.2: Effect of SPPP weighting coefficient on targets found (%) ↑

Parameters		**In-domain**						**Out-domain**					
		$\mathcal{B} = 256$			$\mathcal{B} = 384$			$\mathcal{B} = 256$			$\mathcal{B} = 384$		
β	γ	***All***	***Bot. 5%***	***Bot. 2%***	***All***	***Bot. 5%***	***Bot. 2%***	***All***	***Bot. 5%***	***Bot. 2%***	***All***	***Bot. 5%***	***Bot. 2%***
⅑	2	**57.4**	**28.0**	27.3	**76.1**	**53.0**	**51.9**	**60.8**	**31.7**	**30.7**	**79.6**	**58.9**	**56.1**
⅑	1	57.0	27.6	**27.9**	75.2	51.3	51.9	60.3	29.6	25.7	78.8	54.9	51.4
⅑	0	57.1	26.9	23.9	75.7	50.3	48.8	59.7	27.7	22.6	78.6	53.5	50.2
1	2	56.3	27.0	25.8	75.0	50.5	47.9	60.0	30.2	27.5	78.4	55.9	51.8
–	–	56.6	26.6	27.3	75.5	49.9	51.4	58.5	23.1	16.0	77.1	44.8	36.1

own cluster with the closest alternative cluster. On the other hand, the elbow method charts the within-cluster sum-of-squares across candidate k values and pinpoints where additional clusters yield only marginal variance reduction. Combining both methods balances the silhouette score's tendency to favor fewer clusters with the elbow method's subjectivity based on a 'knee' point. In practice, we set the max averaged k to be 4 based on the approximate number of possible broad semantic landmarks, should the elbow method overestimate k too significantly.

B.5 Varying SPPP-based Online Adaptation Hyperparameters

We perform grid search to determine the optimal hyperparameters for our negative weighting coefficient $\alpha_{\text{neg,j}} = \min\left(\beta\left(O_r/L_r\right)^{\gamma},\, 1\right)$, where O_r is the number of patches observed in region r and L_r is the number of patches in that region. β balances the relative weightage between positive and negative measurements in the loss function, while γ scales the weightage of negative measurements given the same amount of region explored. We summarize our results in Table B.2. When we remove the negative weighting coefficient ($\gamma = 0$), we observe poor performance. This is because all negative samples are weighted equally heavily even at the start of AVS, causing premature collapsing of probability distribution modes. Hence, this highlights the importance of our uncertainty weighting scheme. In addition, if we remove the relative weighting factor ($\beta = 1$), we note one of the worst performances possibly due to over-penalizing negative measurements.

In order to achieve stable updates to the output probability distribution, we reset the satellite encoder weights back to the base weights before running TTA updates. During TTA updates, we use the Adam optimizer, and employ a learning rate schedule that increases our learning rate from 1e-6 to 1e-5 depending on how much of the search space has been covered. This learning rate schedule allows the model to learn more effectively when more measurements are collected [94].

C Additional Baseline Details

In this section, we provide additional information on how we set up our baselines for fair comparison, on top of the details provided in Sec. 6.

C.1 Planner Baselines

We compare Search-TTA with an Attention-based Reinforcement Learning (RL) planner [78] and a greedy Information Surfing (IS) planner [77]. The RL planner is non-myopic in nature as it learns dependencies at multiple spatial scales across the entire search domain. This allows agents to balance trade-offs between short-term exploitation and long-term exploration given the probability distribution map. On the other hand, the IS planner drives agents in the direction of the derivative of the information map to maximize short-term gains. Such an approach tends to be greedy in nature and may suffer from overexploitation of local maxima. By design, the RL planner allows movement to all eight neighboring cells, while the IS planner is limited to the four cardinal directions. Note that we do not intend to compare the performance between RL and IS, but rather how test-time adaptation improves each of the planners independently. Lastly, we use a lawnmower planner [85] as a weak baseline for comparison. Starting from the top-left grid, the lawnmower planner moves in a zigzag

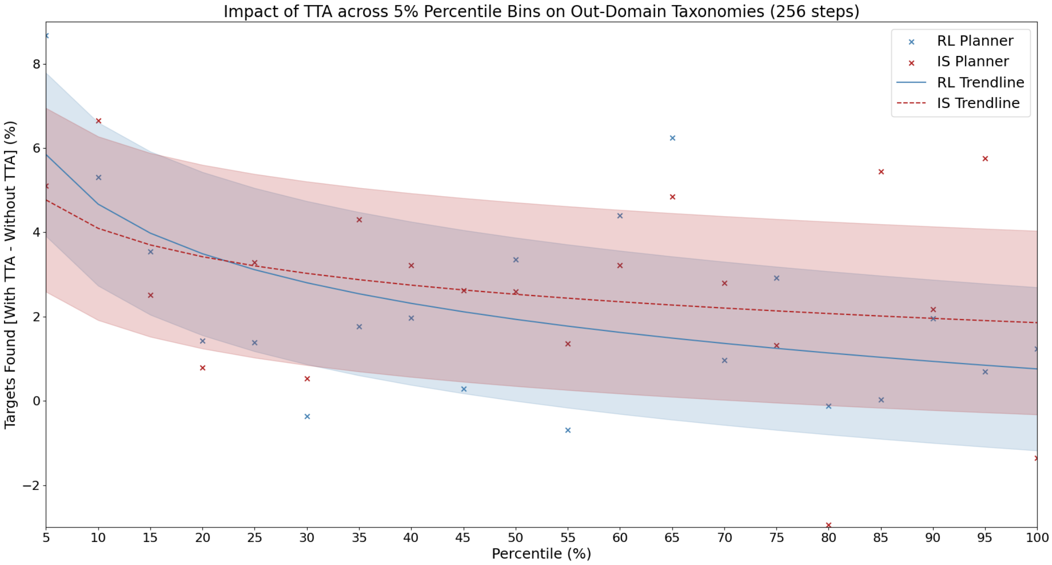


Figure C.1: Performance difference (due to TTA) for RL (blue) and IS (red) planners at 256 steps.

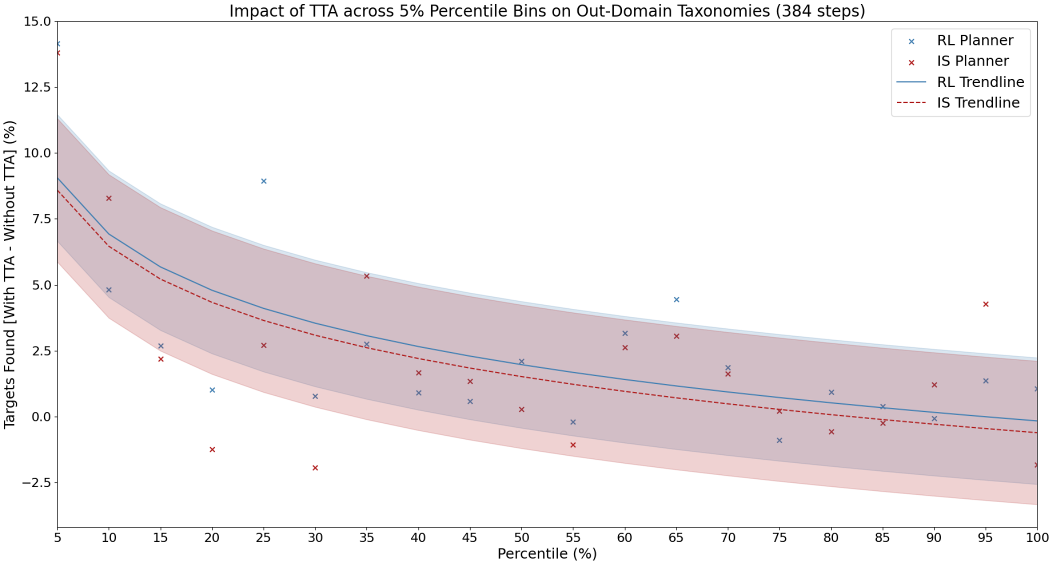


Figure C.2: Performance difference (due to TTA) for RL (blue) and IS (red) planners at 384 steps.

manner across each row before moving down to the next row when the current row is completely covered. This approach, while not using any probability distribution to direct the coverage process, provides an upper bound to the number of cells that can be covered given a specified budget.

RL Policy Training: We train our RL planner's attention-based neural network using the soft actor-critic (SAC) algorithm [95], which learns a policy by maximizing return while maximizing entropy.

$$\pi^* = \operatorname{argmax} \mathbb{E}_{(o_t,a_t)}[\sum_{t=0}^{T} \gamma^t (r_t + \alpha \mathcal{H}(\pi(.|o_t)))], \tag{C.1}$$

where $\mathcal{H}$ denotes entropy, π^* the optimal policy, γ the discount factor, and α the adaptive temperature term. We utilize a subset of the score maps of varying probability distributions from Sec. 4 to pre-train our RL policy. Similar to [37, 78], we define the viewpoints ψ in the search domain $\mathcal{M}$ as graph vertices, each connected via edges to its adjacent (≤ 8) neighbors. In addition to positional information, each of these nodes are augmented with the agent's visitation history and scores from Search-TTA's output probability distribution. This graph can then be used as the RL agent's observa-

Table C.1: Scaling up dataset size for fine-tuning LISA VLM [24] - targets found (%) ↑

VLM	In-Domain		Out-Domain	
	$\mathcal{B} = 256$	$\mathcal{B} = 384$	$\mathcal{B} = 256$	$\mathcal{B} = 384$
LISA (80k)	**57.4**	**76.9**	**60.8**	77.8
LISA (55k)	57.0	76.8	59.0	**78.1**
LISA (24k)	55.2	75.9	56.6	76.8
LISA (no fine-tune)	54.0	72.6	57.9	75.1

tion to output a stochastic policy $\pi_\theta(a^t|o^t)$. Our reward function penalizes distance travelled while incentivizes agents to to travel to high probability regions. We find that adding rewards for targets found causes the planner to generate inefficient routes, possibly due to the sparse target distributions that may provide confusing reward signals. We train our policy using an AMD Ryzen threadripper 3970x and four NVIDIA A5000 GPUs, which took 20k episodes (80 hours) to converge.

Bottom Percentile Comparison: To determine Search-TTA's effectiveness given poor CLIP predictions, we break down the percentage of targets found into the bottom 5% and 2% percentiles. We measure the quality of CLIP predictions by taking the average scores of the pixels where the targets are located on the predicted score map, and deem a CLIP prediction to be poor if most targets are located in the lowest-scoring regions. We then plot the performance gains (due to TTA) for both RL and IS planners for both 256 (Fig. C.1) and 384 steps (Fig. C.2), given bins of 5-percentile increments. Note that the logarithmic trendlines fit the datapoints well, indicating that the performance difference is most significant at the bottom percentiles. This illustrates how Search-TTA is particularly effective given poor score maps in the low percentile range.

C.2 VLM Baselines

We evaluate the effectiveness of Search-TTA's CLIP vision backbone by replacing it with different state-of-the-art VLMs. Most of these VLMs comprise a reasoning module (e.g. LLaVA [22] or Qwen2-VL [87]) that processes image and text inputs to output reasoning embeddings. These embeddings are then passed into segmentation modules such as SAM [80] to generate the appropriate masks. LISA [24] (we use LISA-7B) connects and fine-tunes LLaVA and SAM modules in an end-to-end manner. Unlike the original setup where LISA is trained on binary masks, we directly fine-tune LISA with the **80k** score maps (with text-based question-answer) from our custom training dataset. In addition, we remove the final threshold layer in the SAM module to output continuous score distributions. On the other hand, LLM-Seg [86] (which uses LLaVA-7B) decouples the modules by initially assigning SAM the task of producing mask proposals, which are then evaluated and chosen by LLaVA. Unlike LISA, since there are no straightforward methods to fine-tune LLM-Seg with continuous score maps directly, we apply a binary threshold to our score maps for training (without text-based question-answer). We also incorporate the scores generated by LLM-Seg into all binary mask proposals to output a score map. Lastly, we introduce two fully decoupled baselines that output landmark names and scores from LLaVA-13B and Qwen2-VL-7B. These landmark names are then fed as input into GroundedSAM [81] to generate segmentation masks, which are then aggregated with the VLM's per-region scores to obtain score masks. All of these score maps are passed into the RL policy for path planning [96]. Note that unlike LISA and LLM-Seg, we do not fine-tune LLaVA and Qwen2-VL. The prompts used are shown in Appendix E.2.

Scaling Up VLM Training Dataset: To justify the score map dataset size of **80k**, we experiment with varying the amount of training data used to fine-tune our strongest vision model baseline (LISA-7B) . As seen from Table C.1, scaling up the dataset generally improves search performance. However, the performance gain becomes more marginal when we scale our dataset from 55k to 80k. This indicates generating additional data may not yield further performance gain. Hence, it is a reasonable choice to stop at a dataset size of **80k** (also due to the cost of running GPT4o).

Scaling Up CLIP Dataset: We measure the performance of our satellite image CLIP encoder fine-tuned with data sets of different sizes in Table C.2, to justify why we choose to use the full data set of **380k** images. In particular, we fine-tune CLIP with images from the full **380k** dataset, from the

Table C.2: Scaling up dataset size for fine-tuning CLIP [74] - targets found (%) ↑

Dataset		In-domain						Out-domain					
		$\mathcal{B} = 256$			$\mathcal{B} = 384$			$\mathcal{B} = 256$			$\mathcal{B} = 384$		
Size	**TTA**	***All***	***Bot. 5%***	***Bot. 2%***	***All***	***Bot. 5%***	***Bot. 2%***	***All***	***Bot. 5%***	***Bot. 2%***	***All***	***Bot. 5%***	***Bot. 2%***
380k	Y	57.4	**28.0**	27.3	76.1	**53.0**	51.9	60.8	31.7	**30.7**	79.6	58.9	**56.1**
380k	N	56.6	**26.6**	27.3	75.5	**50.0**	51.4	58.5	23.1	**16.0**	77.1	44.8	**36.1**
200k	Y	56.5	26.0	**22.4**	75.0	49.6	**46.4**	59.8	27.7	**25.1**	79.0	57.0	**55.2**
200k	N	55.6	21.7	**15.2**	74.3	42.5	**32.6**	56.1	20.3	**13.2**	76.1	42.6	**37.2**
80k	Y	53.7	33.0	**30.6**	73.7	59.8	**58.1**	58.7	**36.2**	34.4	78.1	64.0	**62.1**
80k	N	52.8	21.7	**15.6**	72.1	41.9	**38.3**	55.7	**19.9**	18.2	74.9	38.7	**31.5**
No Fine-tune	Y	49.4	**20.9**	16.0	68.4	**47.7**	39.6	50.6	18.7	**18.6**	74.0	53.9	**50.7**
No Fine-tune	N	48.1	**19.6**	17.0	67.8	**36.5**	33.5	49.1	16.3	**11.8**	69.2	34.2	**27.7**

Table C.3: Comparing against prompt learning - targets found (%) ↑

Method	**LR**	**Inference Time (s)**	**In-Domain**		**Out-Domain**	
			$\mathcal{B}$ = 256	**$\mathcal{B}$ = 384**	**$\mathcal{B}$ = 256**	**$\mathcal{B}$ = 384**
Weights (Ours)	(1e-6, 1e-5)	0.15	**57.4**	**76.1**	**60.8**	**79.6**
Prompt [68]	(1e-3, 1e-2)	0.05	57.1	75.3	59.7	78.5
No TTA	–	–	56.6	75.5	58.5	77.1

200k dataset downsampled from the full dataset, and from the **80k** AVS dataset (used to generate score maps for VLM fine-tuning). In addition, we conduct a study in which we do not fine-tune the CLIP model at all. We note an increasing trend in search performance as we scale the dataset. We use a larger training dataset for our CLIP baselines compared to our VLM baselines as these VLMs already have the added advantages of using CLIP as a foundation and being pretrained on much larger datasets. We observe a general increase in TTA performance gain when trained on less data. In particular, we achieve a TTA performance gain of up to 30.0% when trained on the 80k dataset ($\mathcal{B} = 384$, bot. 2%). This indicates Search-TTA's ability to significantly improve score maps when using models trained on less data.

C.3 AVS Framework Baselines

We evaluate the effectiveness of the Search-TTA framework by comparing its performance with existing AVS baselines in the remote sensing domain. Similar to our setup, VAS [17] and PSVAS [16] model an AVS problem where an agent, guided by aerial imagery and operating under a fixed query budget, aims to maximize the number of targets found. VAS utilizes end-to-end reinforcement learning to co-train a feature extraction network and a policy network. The detection results gathered during the search process are then piped back into the feature extraction network for prediction updates. On the other hand, PSVAS decouples its prediction module from its policy network. PSVAS pretrains its prediction module using supervised learning and jointly optimizes both modules using reinforcement learning. During test time, it uses detection results from the search process to directly update the weights of their prediction module. Note that their vision backbones (ResNet [97]) are not foundation models and must be trained on specific classes. In addition, both methods do not perform realistic path planning, but instead allow for querying of non-adjacent cells (i.e. teleporting). For fair comparison. we retain their ability to choose where it wants to query, but also perform Dijkstra path planning to consider the cells on route to their query locations. We weigh our Dijkstra cost function with a combination of factors, which aims to minimize distance, maximize traveling along paths of high probability (output from their vision module), and avoiding visited cells.

D Additional Experiments and Analysis

In this section, we provide information on additional experiments and ablation studies conducted, to supplement the information in Sec. 6. Unless mentioned otherwise, we discretize the search space to 24×24 grids, randomize start positions that is consistent across different validation runs, and perform TTA updates every 20 steps or whenever targets are found.

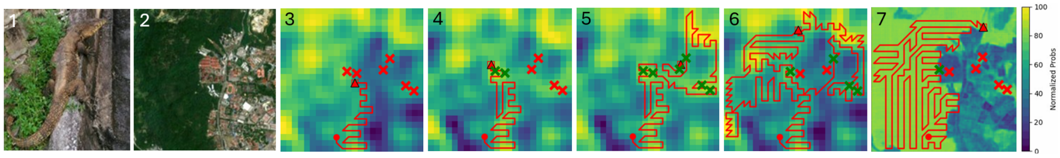


Figure D.1: (1) Ground image of a *Monitor Lizard*. (2) Satellite image where *Monitor Lizards* can be found. (3) Initial CLIP probability output. (4) TTA increases probability values in urban region where the two lizards were found. (5) Improved priors leads to efficient search. (6) Inefficient search without TTA. (7) LISA [24] output probability distribution with higher scores for the forest region, but is unable to perform online adaptation when no targets were found for a prolonged period.

Table D.1: Comparing against text-based TTA (384 Steps) - RMSE (%) ↓

Method	**TTA Type**	***Aves Charadriiformes (Shorebirds)***				
		First (0%)	**Quartile (25%)**	**Mid (50%)**	**Quartile (75%)**	**Last (100%)**
CLIP	SPPP -	**58.7** 58.7	**57.9** 58.7	**56.5** 58.7	**52.4** 58.7	**49.0** 58.7
Qwen2+GroundedSAM [81, 87]	Text -	62.4 62.4	62.6 62.4	60.2 62.4	61.6 62.4	60.9 62.4
LLaVA+GroundedSAM [22, 81]	Text -	59.8 59.8	59.8 59.8	60.2 59.8	58.6 59.8	62.0 59.8

D.1 Varying TTA Methodologies

Prompt Learning: We compare our approach to prompt learning [68], where we perform gradient updates on our satellite image prompt instead of our model weights. From Table C.3, we noticed that a different learning rate range works better for prompt learning, likely due to the number of parameters that are updated during backpropagation as observed in [98]. For fair comparison, we use the same hardware (1x NVIDIA A5000 GPU) to log the inference time. From our results, we observe that weights fine-tuning achieves better averaged performance but has a slower inference time. While the number of parameters updated for prompt learning is significantly less, we only observe three times faster in inference speed, likely due to overheads with PyTorch's computational graphs. We leave comparison with other fine-tuning methods such as LoRA [99] to future works.

Text-based TTA: We study the effect of an alternative text-based TTA strategy [64] aside from our SPPP-based strategy. Instead of integrating the VLMs into our Search-TTA framework which may be time-consuming to test, we design a simple experiment using the region-based statistics logged during our Search-TTA's search process (for *Aves Charadriiformes* / *Shorebirds*). For each region defined by kmeans clustering, we logged the number of targets found and the ratio of the number of patches explored, at the 25%, 50%, 75%, and 100% search checkpoints. We pass these statistics, along with their initial landmark and score predictions (at 0% checkpoint), into the VLM and prompt them to reconsider their predictions. From Table D.1, we note the inconsistency in RMSE values throughout the different checkpoints, in contrast to the consistent improvements made with our SPPP-based strategy. This highlights the importance of a principled approach to TTA to achieve consistent results. We share the prompt design in Appendix E.2.

D.2 Additional Baseline Comparisons

Search-TTA (*In-Domain* Taxonomies): In addition to the experimental results presented in Sec. 6.1 (where $\mathcal{B} = 256$), we present results for $\mathcal{B} = 384$ in Table D.2. Similarly, our results show a general improvement in percentage targets found (especially in the bottom percentile), speed of locating the first target, and quality of score maps generated (as measurements are collected during the search process). This highlights Search-TTA's consistency in improving search performance.

Varying Vision Model: We provide more insights to the data presented in Sec. 6.2 (Table 2). We note the significantly longer inference speed for the Qwen2-VL+GroundedSAM and

Table D.2: Evaluating TTA on different planners (CLIP vision model), on *In-domain* taxonomies

Planner Type	$\mathcal{B} = 256$							$\mathcal{B} = 384$						
	Found (%) ↑			**RMSE (%) ↓**			**Steps ↓ (First tgt)**	**Found (%) ↑**			**RMSE (%) ↓**			**Steps ↓ (First tgt)**
	All	***Bot. 5%***	***Bot. 2%***	***First***	***Mid***	***Last***		***All***	***Bot. 5%***	***Bot. 2%***	***First***	***Mid***	***Last***	
RL (TTA) [78]	57.4	**28.0**	27.3	54.4	53.7	**51.0**	85.1	76.1	**53.0**	51.9	54.4	53.0	**45.9**	101.9
RL (no TTA) [78]	56.6	**26.6**	27.3	54.4	54.4	**54.4**	84.8	75.5	**49.9**	51.4	54.4	54.4	**54.4**	102.7
IS (TTA) [77]	52.1	28.1	**22.1**	54.4	53.4	**51.3**	90.5	71.0	45.0	**40.8**	54.4	52.9	**46.2**	109.6
IS (no TTA) [77]	51.2	21.8	**14.9**	54.4	54.4	**54.4**	90.8	70.2	39.7	**35.3**	54.4	54.4	**54.4**	111.2
Lawnmower [85]	41.8	–	–	–	–	–	112.9	71.5	–	–	–	–	–	148.3

Table D.3: Comparing AVS frameworks (*Animalia Chordata Reptilia Squamata*)

Frameworks	$\mathcal{B} = 256$			$\mathcal{B} = 384$		
	Found (%)	**Explored (%)**	**Steps (First tgt)**	**Found (%)**	**Explored (%)**	**Steps (First tgt)**
CLIP+RL (TTA)	**60.3**	44.3	86.9	**80.5**	65.6	91.3
CLIP+RL (no TTA)	55.9	44.3	78.7	76.9	65.6	95.7
PSVAS [16]	47.3	43.1	83.7	70.0	62.1	109.0
VAS [17]	46.8	44.1	**75.5**	68.5	64.6	**89.3**
Lawnmower [85]	43.5	**44.4**	116.0	71.4	**66.7**	149.1

LLaVA+GroundedSAM baselines. This is because, unlike the other VLMs, Qwen2-VL and LLaVA are required to output landmark names and scores in text, which involves significantly more token generation compared to the custom *[SEG]* token used in LISA and LLM-Seg. LLaVA is slower compared to Qwen2-VL because we use a LLaVA-13B model compared to the Qwen2-VL-7B model. Note that we take all inference speed measurements on a single NVIDIA A5000 GPU.

In addition, we provide snapshots of Search-TTA and LISA to compare their performance when searching for *Monitor Lizards* (*reptilia Aquamata Varanidae*). From Figure D.1, we can see that Search-TTA increases probability values in the urban region after collecting positive samples there. This online adaptation results in a more efficient search (6 targets found). In contrast, LISA over-exploits its initial belief where *Monitor Lizards* are more likely to be found in the forest region, and is unable to correct its probability distribution despite many negative measurements (only 1 target found). Note that our approach without TTA results in only 4 targets found.

AVS Baseline: In addition to the experimental results presented in Sec. 6.3 (*Animalia Chordata Aves Charadriiformes*), we present results for the same AVS framework baselines when searching for *Animalia Chordata Reptilia Squamata* in Table D.3. Likewise, we notice the same trend where Search-TTA outperforms almost all baselines in terms of percentage targets found. This highlights Search-TTA's versatility, given that it is able to outperform AVS baselines (pretrained on specific taxonomies) with just a single model. Although Lawnmower outperforms VAS and PSVAS when $\mathcal{B} = 384$, VAS and PSVAS find the first target more quickly by performing a more targeted search.

Experimental Validation: In addition to the details for our AVS hardware-in-the-loop experiments described in Sec. 6.5, we also rescale our Yosemite Valley simulated environment from 865m × 865m to 280m × 280m. This is due to the crazyflie's limited flight time of 5 minutes, which is not indicative of larger drones that often have longer battery life. The simulated drone flies at an altitude of 30m and is equipped with a 90° FOV camera mounted on a gimbal (tilted at 30° to achieve better bear detection rates). We conduct both experiments, with and without TTA, using $\mathcal{B} = 300s$ (traveled 2058m and 2062m respectively). We execute TTA every 5 iterations when it is enabled.

E Prompt Engineering

E.1 Score Map Generation Prompts

GPT4o Relabelling Prompt

```
You are an AI visual assistant that can analyze a single satellite image that is very zoomed out
(covering around 2 km over the width and 2 km over the height of the image). You are given the
bounding box of the segmentaions of different regions, represented as (x1, y1, x2, y2) with floating
numbers ranging from 0 to 1. These values correspond to the top-left x, top-left y, bottom-right x,
and bottom-right y. These values correspond to the x and y coordinates. x-coordinates increase from
left to right, and y-coordinates increase from top to bottom. The closer the instance is to the top
edge, the smaller the y-coordinate. Assume that point1 [0, 0] is at the upper left, and point2 [1, 1]
is at the bottom right.

The image contains different regions defined by bounding boxes. In the input data the regions are
numbered from 0 to 3. The region names are: "Urban Area", "Barren", "Water", "Vegetation". Your task
is to verify if the region inside the bounding box seems correct as seen in the image. Generate a
conversation between yourself (the AI assistant) and a user asking about the photo. Verify the region
names for each region given in the input data and tell me if it's name matches visually as seen from
the image. When a region is incorrect then also remap the region name to the correct landmark name.
Your responses should be in the tone of an AI that is "seeing" the image and answering accordingly.

When using all the provided information, directly generate the conversation. Always answer as if you
are directly looking at the image.

Mapping:
{
    "Region 0": "Urban Area",
    "Region 1": "Barren",
    "Region 2": "Water",
    "Region 3": "Vegetation"
}

You must return your response in the JSON format:
{
    "conversation": [
        {
            "from": "human",
            "sat_key": sat_key,
            "taxonomy": taxonomy,
        },
        {
            "from": "gpt",
            "landmarks": {
                "Region i": "Correct/Incorrect",
                "Region j": "Correct/Incorrect",
                "Region k": "Correct/Incorrect",
            },
            "corrected_landmarks": {
                "Region i": {
                    "name": "Urban Area/Barren/Water/Vegetation",
                },
                "Region j": {
                    "name": "Urban Area/Barren/Water/Vegetation",
                },
                "Region k": {
                    "name": "Urban Area/Barren/Water/Vegetation",
                },
            }
        }

    ]
}

{Examples}

{Input_Data}

Double check if the region names are correct. In the answer if the region name seems incorrect then
remap it to the correct landmark name, else leave it as it is. Once again, please output your
response in the JSON format only.
```

GPT4o Scoring Prompt

You are an AI visual assistant that can analyze a single satellite image that is very zoomed out (covering around 2 km over the width and 2 km over the height of the image). A specific animal/plant location within the image is given, along with detailed coordinates. The locations are in the form of coordinates, represented as (x,y) with floating numbers ranging from 0 to 1. These values correspond to the x and y coordinates. x-coordinates increase from left to right, and y-coordinates increase from top to bottom. The closer the instance is to the top edge, the smaller the y-coordinate. Assume that point1 [0, 0] is at the upper left, and point2 [1, 1] is at the bottom right.

You are also given the name of the animal/plant as taxonomy. Along with the exact coordinates in the image, you are also given the bounding box of the area where the animal/plant was found, represented as (x1, y1, x2, y2) with floating numbers ranging from 0 to 1. These values correspond to the top-left x, top-left y, bottom-right x, and bottom-right y. The values follow the same format as the coordinates.

The image contains different landmarks defined by bounding boxes. If a coordinate lies inside a particular bounding box and that bounding box belongs to a specific landmark, that means the animal/plant was found in that landmark. Your task is to generate a conversation between yourself (the AI assistant) and a user asking about the photo. Your responses should be in the tone of an AI that is "seeing" the image and answering accordingly. Using the image, taxonomy, coordinates, and bounding box information, provide a score for each landmark and a detailed explanation of where the animal/plant could be found among those landmarks based on the landmark semantics from the image. Add a simple question in the conversation, asking about where to find the particular animal/plant in the image (use common name as well as taxonomy for question).

Given the explanation, evaluate a score based on the likelihood of finding the queried animal/plant in each landmark. The length of the 'landmarks' list must match the length of the 'score' list. Probability scores must range between 0.0 and 1.0. It is acceptable for multiple consecutive landmarks to have the same scores. For example, a frog may have the same score for both water and land-type landmarks. Scores can be any value between 0.0 and 1.0, such as 0.1, 0.3, 0.5, 0.7, or 0.9. You must be more conservative, where if you are not sure, you should assign a lower score. If the animals cannot exist inside this landmark, please assign a score of 0.0. For example, a land animal or a non-aquatic plant cannot live inside the water body.

The scoring system is defined as follows:

1.0: Almost guaranteed to find the animal in the landmark
0.8: Very likely to find the animal in the landmark
0.6: Likely to find the animal in the landmark
0.4: Unlikely to find the animal in the landmark
0.2: Very unlikely to find the animal in the landmark
0.0: Almost impossible to find the animal in the landmark

You must answer with an explanation, landmarks, and scores for each region. Score the area of a particular region not only based on the number of targets actually present but also using the semantic information of the region. Provide a detailed explanation for your scoring, describing the relationship between the region's semantics and the animal/plant.

When using all the provided information, directly generate the conversation. Always answer as if you are directly looking at the image. Do not mention bounding box or region numbers explicitly, instead use the assigned landmark names. Only output the landmark name, not the landmark coordinates. Your answer may include multiple landmark types. It is acceptable for multiple consecutive landmarks to have the same scores.

You must return your response in the JSON format:

```
{
    "conversation": [
        {
            "from": "human",
            "sat_key": sat_key,
            "taxonomy": taxonomy,
            "common_name": common_name,
            "question": question using taxonomy,
        },
        {
            "from": "gpt",
            "explanation": answer,
            "landmarks": {
                "landmark1": {
                    "score": score,
                },
                "landmark2": {
                    "score": score,
                },
                "landmark3": {
                    "score": score,
```

````
                },
            }
        }
    ]
}

{Examples}

{Input_Data}

Once again, please output your response in the JSON format.
````

E.2 VLM Baselines Prompts

Llava Inference Prompt (LLaVA+GroundedSAM Baseline)

````
Using the image as a reference, where can {animal} be found? Give me 1-2word high-level landmark
names where it can be found in the image. Your response will be a json object where the landmark
names are the keys and the probability of it being found in the landmark as values e.g
{{"barren_land": 0.6}}. Return just between 3 to 5 landmarks. Your response must be a single json
object enclosed in double quotes without additional text.
````

Llava TTA Prompt (LLaVA+GroundedSAM Baseline)

````
You are provided with a heat map of the satellite image earlier, region statistics showing the
percentage of the region in the satellite image that has been explored and number of {animal} found
in these regions. Region Statistics:\n{explore_info}\n Use these information to update the
probabilities of {animal} being found in the landmarks generated previously: {orig_response}. Do not
associate the region Rn with any of the landmarks, they are not related in any way. Return your
answer as a JSON object in this format: {new_dict}, where the keys are enclosed with double quotes.
Begin your answer with explanation and reasoning steps for calculating the new probability values for
{landmark_names}, then return the JSON object. You must enclose the JSON object within ```json tag.
e.g '''json{{"<landmark>": <new value>}}```.
````

Qwen Inference Prompt (Qwen2+GroundedSAM Baseline)

````
Using the image as a reference, where can {animal} be found? Give me 1-2word high-level landmark
names where it can be found in the image. Your response will be a json object where the landmark
names are the keys and the probability of it being found in the landmark as values e.g
{{"barren_land": 0.8}}. Return just between 3 to 5 landmarks. The key should be landmark names, not
any other animals or food. Your response must be a single json object enclosed in double quotes
without additional text, and do not return the examples given as a result.
````

Qwen TTA Prompt (Qwen2+GroundedSAM Baseline)

````
You are provided with a satellite image,a heat map of the satellite image and region statistics and
tasked to use these information to come up with pairs of landmark:probabilities, where probabilites
denotes the chances of finding {animal} in the landmark. The heat map, together with region
statistics shows the percentage of the region in the satellite image that has been explored, as well
as the number of {animal} found in it. Region Statistics:\n{explore_info}\nSuppose the original
response {orig_response}, you need to use the heat map and region statistics information to come up
with the new probabilities associated with the given landmark, using the satellite image as a
reference. Do not associate regions Rn with {landmark_names}, they are not related to one another.
Return your answer in this format: {new_dict} with new probability values (between 0 and 1) you
calculated from the region statistics. Begin your response with explanation and reasoning steps for
calculating new values for {landmark_names}, and return a single JSON object in {new_dict} format
with the new probability values, enclosed in double quotes for the key and values. Enclose the json
object within the ```json tag. e.g ```json{new_dict}```. The JSON key should be enclosed by double
quotes.
````