ENHANCING AERIAL VISION-LANGUAGE NAVIGATION WITH MAP GROUNDING AND HISTORY AWARENESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language Navigation (VLN) for urban UAVs is frequently hindered by "landmark blindness," where target landmarks are not visible from the agent's initial viewpoint. We address this by fine-tuning small Vision-Language Models using a "Map-in-Pixel" approach that interleaves 16 steps of egocentric visual frames with global geographic snapshots. To mitigate the data scarcity inherent in VLN datasets, we propose a synthetic augmentation strategy that generates diverse, causally consistent trajectories from randomized starting points. Through granular evaluation and targeted trajectory synthesis, we demonstrate that this history-rich training significantly improves the agent's ability to navigate toward distant objects. Our approach achieves a success rate of 12.5% on the CityNav unseen test set, nearly doubling the baseline (6.4%), while simultaneously reducing navigation error below baseline levels. This work underscores the efficacy of pixel-encoded maps, temporal history, and targeted data-centric design in empowering small-scale multimodal agents for long-horizon missions.

1 INTRODUCTION

Recent progress in vision–language models (VLMs) has enabled agents to follow natural language instructions in complex visual environments. Extending VLN to unmanned aerial vehicles (UAVs) introduces additional challenges: 3D motion, long-horizon planning, partial observability, and increased sensitivity to control artifacts in the data. City-scale aerial benchmarks such as CityNav (Lee et al., 2024) provide realistic environments reconstructed from aerial imagery and point clouds, along with human-piloted trajectories. However, effectively training VLM-based policies on CityNav remains difficult: baseline models struggle to interpret the landmark map reliably and can be dominated by spurious biases such as skewed action distributions.

In this work we fine-tune a compact open-source VLM for map-grounded aerial navigation on CityNav. The model receives three inputs at each step: a natural language instruction, a landmark map showing the agent pose and landmark polygon, and an egocentric RGB view. It then autoregressively predicts the next discrete action token.

We make three contributions:

- **Action imbalance diagnosis and mitigation.** We show CityNav has a severe action skew (e.g., forward/down dominate), which biases token-level action prediction. We mitigate this via *action grouping*, introducing composite action tokens that reduce distribution skew.
- **History-aware conditioning.** We incorporate recent trajectory context by providing a sequence of past landmark maps. This substantially improves generalization on Val Unseen.
- **Skill-focused synthetic supervision.** To address overfitting and the recurring failure mode of misreading the map, we propose a targeted synthetic dataset covering discrete geometric cases for landmark/target alignment, improving navigation error on Test Unseen while preserving success.

2 Related Work

Aerial VLN benchmarks. Aerial Vision-and-Dialog Navigation (AVDN) (Fan et al., 2022) introduces dialog-driven UAV control in a photorealistic simulator, emphasizing interactive instruction following. AerialVLN (Liu et al., 2023) proposes city-scale UAV VLN in a 3D simulator with near-realistic renderings and height-aware control. OpenFly (Gao et al., 2025) scales aerial VLN with a large toolchain and multi-granularity actions. CityNav (Lee et al., 2024) occupies a complementary regime: human-piloted trajectories rendered from real imagery and 3D point clouds, enabling AirSim-based photogrammetric simulation (Shah et al., 2018).

Map-grounded navigation with VLMs. Recent systems explore reasoning over global semantic maps and structured representations for UAV navigation (e.g., FlightGPT (Cai et al., 2025)), often under a formulation where the model has broad global context. In contrast, CityNav provides a local landmark map and egocentric view; we show that in this local-observation regime, data-centric interventions (action grouping, history, and targeted synthetic skills) yield strong gains without requiring complex global-map reasoning. We note that several recent UAV systems operate in a different problem setup where the model receives a global map (or the entire scene map) and directly predicts the destination or waypoint, rather than producing step-by-step navigation actions. Since our setting requires sequential action prediction under local observations, such global-map formulations are not directly comparable to CityNav action-based baselines, and we treat them as complementary.

3 CityNav Dataset

We use CityNav (Lee et al., 2024), which provides instruction-goal navigation trajectories in environments reconstructed from real aerial imagery and point clouds. At each step, the agent observes a top-down landmark map with pose/orientation and an egocentric RGB view. The goal is to follow the instruction and reach the target; we evaluate using Navigation Error (NE, lower is better), Success Rate (SR, higher is better), and Oracle Success Rate (OSR, higher is better) under the benchmark protocol. The camera observes only a local neighborhood around the UAV, while a full town map with landmarks is available as the landmark map input.

4 Method

4.1 Policy Model

We fine-tune Qwen3-VL-4B (Bai et al., 2025) as the base model. At each step, the input consists of: (i) the instruction text, (ii) the current visual observations: landmark map image (with landmark polygon and agent pose/orientation) and first-person RGB view. (iii) a serialized history of past landmark maps, first-person RGB views and previous actions. All visual inputs are processed by the same frozen vision encoder. We train with next-token prediction using cross-entropy loss. We represent each action as a short natural-language token (e.g., `[Forward]`, `[Left]`, `[Right]`, `[Up]`, `[Down]`, `[Stop]`), enabling the VLM to learn action prediction in the same autoregressive decoding space as standard text generation.

4.2 Training Details

We use a peak learning rate of 2×10^{-5} with cosine scheduling and warmup. We train with batch size 4 per GPU across 8 NVIDIA H100 GPUs. We resize the landmark map to 112×112 and the local view to 224×224 for efficiency while retaining spatial cues. We freeze the vision encoder in all experiments.

4.3 Action Grouping for Imbalance

CityNav exhibits a strong action imbalance: MOVE FORWARD and GO DOWN appear far more frequently than other actions, which biases token-level learning. Following the intuition in OpenFly (Gao et al., 2025), we introduce composite action tokens that represent repeated primitives: GO UP$\times$4, GO DOWN$\times$4, and MOVE FORWARD$\times$2. For example, four consecutive 2m upward steps

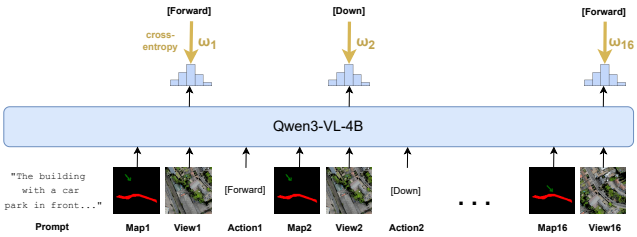


Figure 1: Overview of the method pipeline. The model receives an instruction, a landmark map (with pose/orientation), and an egocentric RGB view, and autoregressively predicts discrete action tokens.

become a single 8m upward token; two consecutive 5m forward steps become a 10m forward token. This reduces distribution skew and improves generalization by discouraging over-prediction of dominant primitives.

4.4 History Conditioning

Navigation decisions benefit from awareness of past observations and actions. We condition the model on a sequence of past landmark maps (history size $H \in \{0, 8, 16\}$) by concatenating them into the input context. We tune H on Val Unseen and use H=16 for subsequent experiments.

History reset. During inference, episodes often exceed the history window size H. Rather than maintaining a sliding window of the most recent H maps, we reset the history buffer to empty once H steps have been accumulated. This prevents stale context from degrading predictions in long episodes and allows the model to re-orient from a clean state periodically.

4.5 Loss Variants

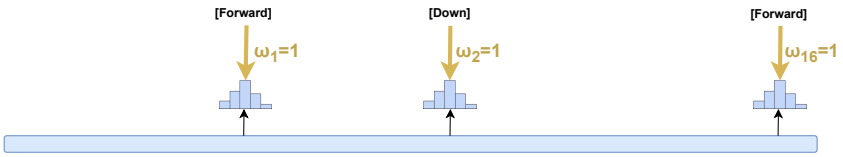
We investigate several loss formulations to better align training objectives with the navigation task. In our setting, the model autoregressively predicts a sequence of tokens, where the *final action token* determines the actual navigation behavior executed by the UAV. Consequently, correctness of the last predicted token is particularly critical, as it directly corresponds to the chosen navigation action.

We experiment with three loss variants:

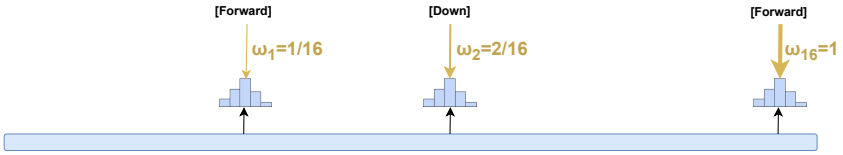
Baseline loss. Our baseline uses the standard autoregressive cross-entropy loss, where all tokens in the output sequence are weighted equally. This corresponds to the conventional training objective used in most Vision–Language Model fine-tuning setups and serves as a reference point.

Linearly weighted loss. To emphasize later tokens in the output sequence, we introduce a weighted cross-entropy loss where token weights increase linearly from 0 to 1 across the sequence. Formally, earlier tokens receive lower weights, while tokens closer to the end of the sequence are assigned higher importance. This formulation biases learning toward accurately predicting later tokens, while still retaining supervision over the entire sequence.

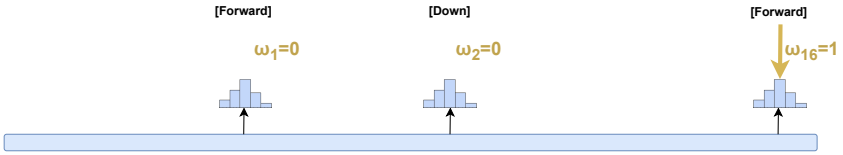
Last-token-only loss. We also experiment with a loss that is applied *only* to the final token of the output sequence. In this formulation, intermediate tokens are ignored during loss computation, and optimization focuses exclusively on predicting the final action token correctly. The motivation behind this variant is that navigation correctness is ultimately determined by the last predicted action, making it a direct proxy for decision accuracy.



(a) Baseline loss (all tokens equally weighted)



(b) Linearly weighted loss (increasing weight toward final token)



(c) Last-token-only loss

Figure 2: Comparison of different loss formulations used during training. (a) Standard cross-entropy applied uniformly across all tokens. (b) Linearly weighted loss emphasizing later tokens in the action sequence. (c) Loss applied only to the final token, which directly determines the next navigation action.

Motivation. These loss variants are designed to explore the trade-off between full-sequence supervision and decision-focused optimization. By emphasizing or isolating the final token, we aim to better align the training objective with the downstream navigation behavior, where the final action prediction determines the UAV's movement. We evaluate these loss formulations in terms of navigation accuracy and overall task performance in Section 6.

Last-token validation. In addition to modifying the training loss, we align the validation protocol with the decision-making nature of the navigation task. Specifically, validation metrics are computed only on the final predicted token, which corresponds to the action executed by the UAV. Since navigation performance is determined by this final action choice, intermediate tokens provide limited signal about control correctness. We observe that last-token validation better correlates with downstream navigation metrics, as errors in the final action prediction can significantly affect trajectory outcomes.

5 Synthetic Dataset

A key challenge we observed was significant overfitting when training on the CityNav dataset alone, likely due to its limited size and diversity for fine-tuning a large Vision–Language Model. As shown in Figure 3, the training loss consistently decreases while the validation loss stagnates, indicating that the model is memorizing the training data rather than learning generalizable navigation skills.

To mitigate this and enhance the model's ability to perform the navigation task, we generated a synthetic dataset to support fine-tuning. This dataset was specifically designed to teach fundamental navigation concepts and to improve generalization across different spatial configurations.

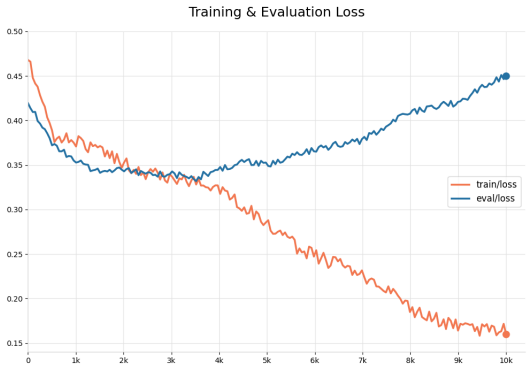


Figure 3: Training and validation loss curves. The divergence between the two curves indicates significant overfitting on the CityNav training data after several epochs.

We categorize the drone's position and corresponding action into six classes:

C1: Far from the landmark, random arrow direction. The drone is positioned far from the landmark, and the direction arrow is random. The model should predict a corrective action—turning left or right—based on the landmark's relative position.

C2: Far from the landmark, arrow towards the landmark. The drone is far from the landmark, but the direction arrow is correctly pointing towards it. The expected action is to move forward.

C3: Near the landmark, target not visible. The drone is close to the landmark, but the target is no longer visible. This is ambiguous with no definitive correct next step, so we do not generate synthetic data for this class.

C4: Near the landmark, target visible, random arrow. The drone is near the landmark, the target is visible, but the arrow direction is random. The correct action is to adjust orientation by turning left or right.

C5: Near the landmark, target visible, arrow towards the target. The drone is close to the landmark, the target is visible, and the arrow points towards it. The drone should proceed forward or downwards, depending on proximity and altitude.

C6: Very close to the landmark and the target. When the drone is within 20 meters of both the landmark and the target, the correct action is to stop.

In its original formulation, the synthetic dataset focused on supervising corrective behaviors (moving forward and turning left/right). To also teach the model to recognize task completion, we add an explicit zero-action (STOP) case representing terminal states where the drone has already reached the target. These samples complement the existing corrective-action cases by supervising when no further movement is required.

We generate synthetic data covering five of the six defined cases; the third scenario is intentionally excluded due to ambiguity. Zero-action (STOP) samples are generated separately based on arrival conditions and included alongside the other supervised cases.

Synthetic case types used in ablations. For compact analysis, we also group synthetic samples into six interpretable geometric cases: landmark-based alignment when far from the landmark ($\Leftarrow L$, $\Rightarrow L$, $\Uparrow L$) and target-based alignment when the target is visible ($\Leftarrow T$, $\Rightarrow T$, $\Uparrow T$). Here, $\Uparrow$ indicates the UAV is oriented towards the relevant goal (landmark or visible target), while $\Leftarrow$ / $\Rightarrow$ indicate the UAV is oriented incorrectly and requires a left/right correction.

We generate two distinct sets of synthetic data: one for training (from CityNav training samples) and one for evaluation (from validation/test samples). Each set is subdivided by action category (forward, left, right, and terminal stop), enabling systematic evaluation of map interpretation and termination recognition. We later analyze the effect of varying the proportion of synthetic data in Section 6.4.

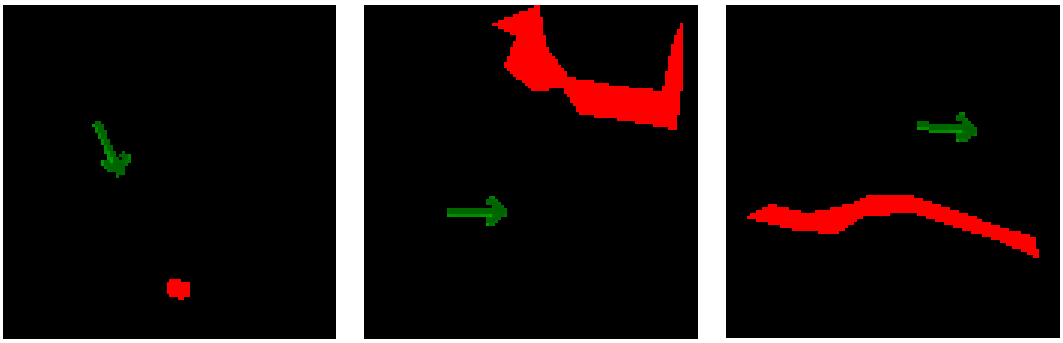


Figure 4: Representative cases from the synthetic dataset. The top row shows **landmark-based** navigation samples, while the bottom row presents **target-based** navigation samples, including a **STOP (0-case)** example where the drone has already reached the target. The correct action for each state is shown below each image. Red rectangles indicate ground-truth target locations, red polygons denote landmarks, and arrows show the drone's current position and orientation.

Table 1: Val Unseen performance vs. history size H.

History Size	NE ↓	SR ↑	OSR ↑
0	190.0	0.82	0.91
8	120.0	6.05	10.90
16	160.0	6.11	14.46

6 Experiments

6.1 Metrics and Splits

We report Navigation Error (NE), Success Rate (SR), and Oracle Success Rate (OSR) on Val Unseen and Test Unseen, including difficulty bins (Easy/Medium/Hard) and aggregate (All), following CityNav evaluation.

6.2 History Ablation

Table 1 shows performance on Val Unseen for different history sizes. Moving from $H{=}0$ to $H{=}8$ yields a large gain in success and oracle success. Increasing to $H{=}16$ yields similar SR with higher OSR, indicating improved reachability awareness but potentially harder sequence processing. We also emphasize that our setup performs *step-by-step* navigation by predicting actions; this differs from approaches that receive the entire map and directly predict destinations/waypoints, which are therefore not directly comparable to CityNav baselines.

Table 2: Synthetic data proportion study on Val Unseen.

Configuration				Val Unseen		
Synth 10%	Synth 20%	Synth 30%	Reset	NE↓	SR↑	OSR↑
✓				132.03	6.38	10.26
	✓			176.39	4.45	8.61
		✓		132.91	6.49	9.68
		✓	✓	**101.30**	**9.43**	**16.21**

Table 3: Action prediction evaluation, before and after augmentation with 30% synthetic data.

	Before Aug.		After Aug.	
Action	Landmark	Target	Landmark	Target
Forward	0.72	0.84	0.96	0.92
Turn Right	0.22	0.35	0.94	0.90
Turn Left	0.32	0.42	0.96	0.92
Stop	—	0.37	—	0.97

6.3 Synthetic Data Proportion Study

We study how the proportion of synthetic samples influences navigation performance. We conduct a grid of experiments in which synthetic data constitutes 10%, 20%, and 30% of the training set, with the remainder drawn from original CityNav trajectories. Table 2 presents the results on Val Unseen. We observe that 10% synthetic data yields strong gains on its own, while 30% synthetic data underperforms without complementary techniques. However, combining 30% synthetic data with history reset (and our chosen loss) yields the best trade-off.

6.4 Synthetic Augmentation Analysis

We evaluate action prediction accuracy on the synthetic evaluation set before and after augmentation with synthetic data (Table 3). Augmentation yields near-uniform improvements across all action types and alignment conditions, with landmark turning accuracy increasing from 0.22–0.32 to 0.94–0.96, and target stop recognition improving from 0.37 to 0.97.

6.5 Loss Variant Analysis

To validate our choice of loss function, we compare the baseline cross-entropy loss against the linearly weighted and last-token-only variants on the validation set, using a fixed 30% synthetic data mixture and history reset (Table 4). While the baseline loss yields reasonable performance, the linearly weighted loss achieves the highest success rate (11.08%) and oracle success rate (29.12%). Based on these results, we select the linearly weighted loss for our final model.

6.6 Main Results

Table 5 compares our full system (H=16, 30% synthetic data, weighted loss, history reset) against the CityNav baseline. On Test Unseen (All), we achieve SR 12.54% vs. 6.38% for the baseline, nearly doubling the success rate, while simultaneously reducing navigation error from 93.84m to 81.82m. On Easy episodes, our model reaches 17.37% SR with NE of 62.60m, compared to 6.15% SR and 98.9m NE for the baseline.

7 Discussion

Our results show that in CityNav, a large portion of performance is governed by data and representation choices rather than model scale. Action grouping mitigates a token-level bias caused by

Table 4: Ablation of loss variants on Val Unseen (All), using 30% synthetic data and history reset.

Loss Variant	NE ↓	SR ↑	OSR ↑
Baseline (Standard CE)	101.30	9.43	16.21
Last-Token-Only	89.97	10.11	17.61
Linearly Weighted	**75.15**	**11.08**	**29.12**

Table 5: Comparison with the CityNav baseline on Test Unseen across difficulty bins.

Split	Model	Easy			Medium			Hard		
		NE↓	SR↑	OSR↑	NE↓	SR↑	OSR↑	NE↓	SR↑	OSR↑
Test Unseen	CityNav Baseline	98.9	6.15	39.89	90.9	6.29	21.47	90.0	6.80	12.10
Test Unseen	Ours	62.60	17.37	30.61	86.02	10.68	15.78	104.18	7.80	10.27

Split (All)	Model	NE↓	SR↑	OSR↑
Test Unseen	CityNav Baseline	93.84	6.38	**26.17**
Test Unseen	Ours	**81.82**	**12.54**	20.14

imbalanced trajectories. History conditioning provides essential context for disambiguating map alignment and avoiding repetitive behaviors. Synthetic supervision is effective when it targets a concrete failure mode (map misinterpretation) using a small set of discrete geometric cases. The higher Success Rate and lower Oracle Success Rate likely originates from excluding the ambiguous C3 case (landmark reached, target invisible) from the synthetic data. This scenario is crucial for learning goal approach behavior. However, the model remained capable of close-range target detection due to training on C6 cases.

8 Conclusion

We fine-tune a compact open-source VLM for map-grounded aerial navigation on CityNav. By addressing action imbalance via action grouping and incorporating recent landmark-map history with an inference-time reset strategy, we improve generalization and achieve a 12.54% SR on Test Unseen under a step-by-step action prediction protocol, nearly doubling the baseline while reducing navigation error below baseline levels. To mitigate overfitting and explicitly teach map interpretation and termination recognition, we introduce a targeted synthetic dataset of discrete geometric cases and an additional zero-action (STOP) case; augmentation substantially improves both navigation error and success rate. These findings suggest that small VLMs can be competitive for aerial VLN when paired with careful data-centric design and skill-focused synthetic supervision.

References

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025. URL https://arxiv.org/abs/2511.21631.

Hengxing Cai, Jinhan Dong, Jingjun Tan, Jingcheng Deng, Sihang Li, Zhifeng Gao, Haidong Wang, Zicheng Su, Agachai Sumalee, and Renxin Zhong. Flightgpt: Towards generalizable and in-

terpretable uav vision-and-language navigation with vision-language models. *arXiv preprint arXiv:2505.12835*, 2025. URL https://arxiv.org/abs/2505.12835.

Yue Fan, Winson Chen, Tongzhou Jiang, Chun Zhou, Yi Zhang, and Xin Eric Wang. Aerial vision-and-dialog navigation. *arXiv preprint arXiv:2205.12219*, 2022. URL https://arxiv.org/abs/2205.12219.

Yunpeng Gao, Chenhui Li, Zhongrui You, Junli Liu, Zhen Li, Pengan Chen, Qizhi Chen, Zhonghan Tang, Liansheng Wang, Penghui Yang, Yiwen Tang, Yuhang Tang, Shuai Tang, Songyi Liang, Ziqin Zhu, Ziqin Xiong, Yifei Su, Xinyi Ye, Jianan Li, Yan Ding, Dong Wang, Zhigang Wang, Bin Zhao, and Xuelong Li. Openfly: A versatile toolchain and large-scale benchmark for aerial vision-language navigation. *arXiv preprint arXiv:2502.18041*, 2025. URL https://arxiv.org/abs/2502.18041.

Jungdae Lee, Taiki Miyanishi, Shuhei Kurita, Koya Sakamoto, Daichi Azuma, Yutaka Matsuo, and Nakamasa Inoue. Citynav: Language-goal aerial navigation dataset with geographic information. *arXiv preprint arXiv:2406.14240*, 2024. URL https://arxiv.org/abs/2406.14240.

Shubo Liu, Hongsheng Zhang, Yuankai Qi, Peng Wang, Yaning Zhang, and Qi Wu. Aerialvln: Vision-and-language navigation for uavs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15338–15348, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/papers/Liu_AerialVLN_Vision-and-Language_Navigation_for_UAVs_ICCV_2023_paper.pdf.

Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In Marco Hutter and Roland Siegwart (eds.), *Field and Service Robotics: Results of the 11th International Conference*, pp. 621–635. Springer International Publishing, Cham, 2018. doi: 10.1007/978-3-319-67361-5_40. URL https://arxiv.org/abs/1705.05065.