NaVILA: Legged Robot Vision-Language-Action Model for Navigation

An-Chieh Cheng[1,*] Yandong Ji[1,*] Zhaojing Yang[2,*] Zaitian Gongye[1] Xueyan Zou[1]
Jan Kautz[3] Erdem Bıyık[2] Hongxu Yin[3,†] Sifei Liu[3,†] Xiaolong Wang[1,3,†]
[1]UC San Diego [2]USC [3]NVIDIA

https://navila-bot.github.io

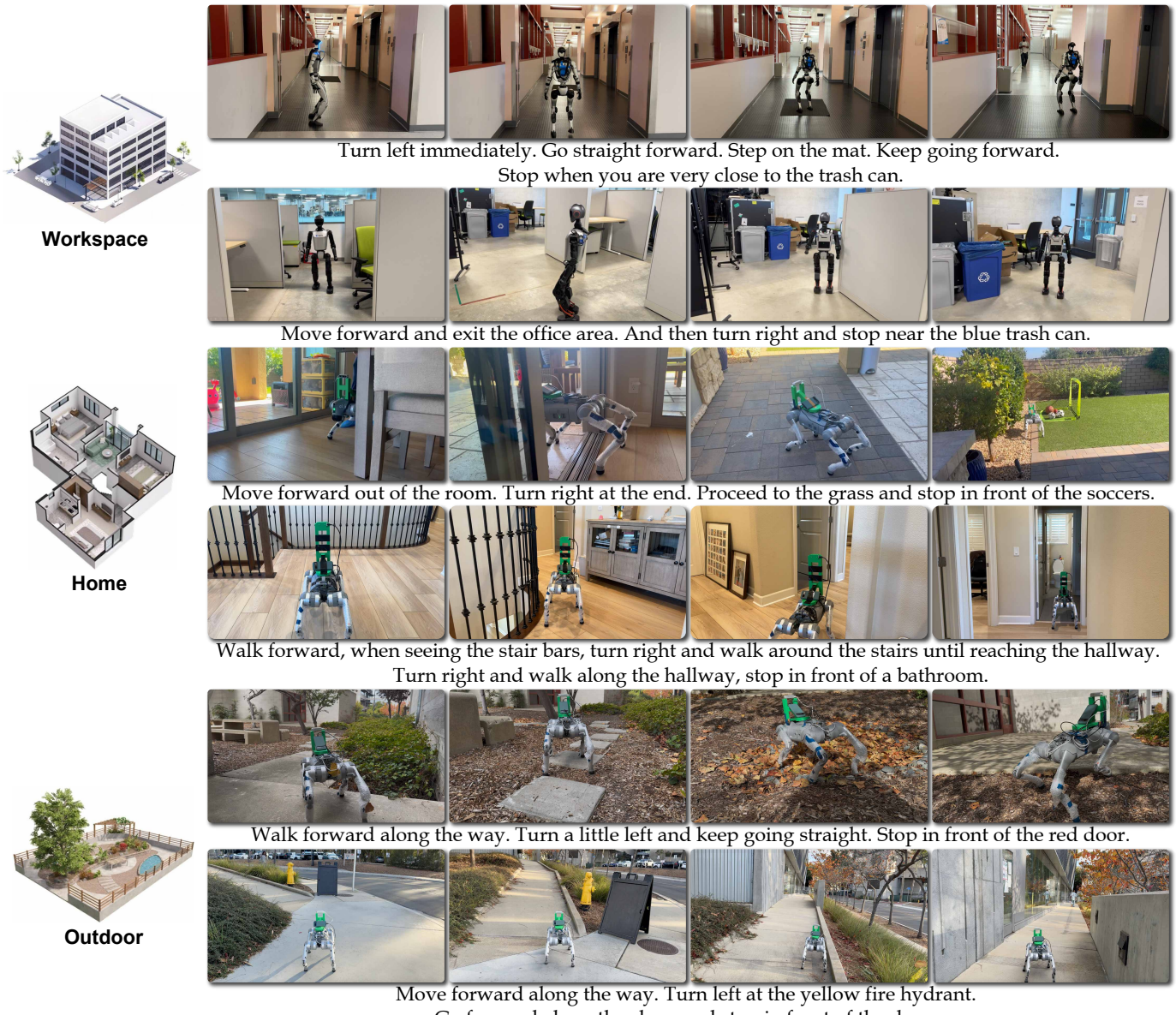


Fig. 1: Real-world demonstration of NaVILA: Upon receiving human instructions, NaVILA uses a vision-language model to process RGB video frames and employs locomotion skills to execute the task on a robot. The robot successfully handles long-horizon navigation tasks and operates safely in challenging environments.

***Abstract*—This paper proposes to solve the problem of Vision-and-Language Navigation with legged robots, which not only provides a flexible way for humans to command but also allows the robot to navigate through more challenging and cluttered scenes. However, it is non-trivial to translate human language instructions all the way to low-level leg joint actions. We propose NaVILA, a 2-level framework that unifies a Vision-Language-Action model (VLA) with locomotion skills. Instead of directly predicting low-level actions from VLA, NaVILA first generates mid-level actions with spatial information in the form of language, (e.g., "moving forward 75cm"), which serves as an input for a visual locomotion RL policy for execution. NaVILA substantially**

∗ Equal contribution, ordered alphabetically. † Equal advising.

improves previous approaches on existing benchmarks. The same advantages are demonstrated in our newly developed benchmarks with IsaacLab, featuring more realistic scenes, low-level controls, and real-world robot experiments.

I. Introduction

The ability to perform Vision-and-Language Navigation (VLN) has become a foundational component in modern robotics systems. With VLN, a robot is expected to navigate around unseen environments without a provided map following a language instruction [1–6]. This not only offers a better interface for humans, but also strengthen cross-scene generalization through languages. In this paper, we further extend the study of VLN with legged robots (e.g., quadruped or humanoid). Using legs instead of wheels allows robots to navigate in more challenging and cluttered scenarios. As the examples shown in Fig. 1, our robot can navigate through a messy laboratory space with narrow walkways, transition from room to room in a house, as well as tackle outdoor challenging environments such as uneven terrains with small rocks, holes, and troughs.

To translate language to action, the robot needs to reason about the input language, and perform closed-loop planning as well as low-level control. With the recent advancement in Large Language Models (LLMs) and Vision-Language Models (VLMs), several end-to-end Vision-Language-Action (VLA) systems have been developed [7–9]. These systems fine-tune a general-propose VLM with large-scale robot manipulation demonstrations to produce low-level actions. While unifying reasoning and execution in a single model is fascinating and shows encouraging results, it is worth diving deeper into the question: Is there a better way to represent actions beyond the quantized low-level commands? After all, LLMs and VLMs were primarily trained with natural language. Unifying reasoning and execution becomes challenging when we need to convert that reasoning into precise, non-verbal actions.

Inspired by the recent progress on VLM [10, 11] for spatial location and distance reasoning, we propose **NaVILA**, a two-level framework for legged robot VLN: A VLM is fine-tuned to output a **mid-level action** (VLA) in the form of language such as "turn right 30 degrees", and a low-level visual locomotion policy is trained to follow this instruction for execution. The mid-level action output of the VLA conveys the location and direction information without the low-level commands. The advantages of this framework are three-fold: (i) By decoupling low-level execution from VLAs, the same VLA can be applied across different robots by swapping the low-level policy; (ii) Representing actions as mid-level language instructions enables VLA training with diverse data sources, including real human videos and reasoning QA tasks. This enhances reasoning capabilities without overfitting outputs to specific low-level commands and can leverage real-world data for generalization; (iii) NaVILA operates on two distinct timescales: the VLA, typically a large and computationally intensive model, runs at a lower frequency, providing high-level navigation commands; while the locomotion policy operates in real-time. This dual-frequency approach allows the locomotion policy to handle sophisticated obstacle avoidance and increases overall robustness.

To train the VLA, we demonstrate how to (i) integrate historical context and current observations in VLN within existing VLM frameworks, (ii) create a specialized navigation prompt tailored for VLN tasks, (iii) utilize real-world data from YouTube human touring videos to improve navigation in continuous environments, and (iv) introduce a carefully curated dataset blend designed to enhance VLN generalizability. These strategies allow us to fine-tune a general-purpose image-based VLM into a navigation-focused agent while simultaneously training it on general vision-language datasets, thereby maintaining its broad generalization capabilities. Moreover, this is the first work to show that direct training on human videos improves navigation in continuous environments.

To train robust locomotion skills, we employ a single-stage approach to learn vision-based locomotion policy. We construct a height map from raw LiDAR point clouds and introduce randomization to bridge the sim-to-real gap. This controller takes the output from our VLA model, converts it into command velocities, and tracks these velocities by controlling the positions of the joints. This end-to-end approach enables the training of visual locomotion skills that are both robust and safe, facilitating deployment in real-world, challenging environments (e.g., strong sunlight or near certain transparent surfaces).

In our experiments, we show that our VLA significantly outperforms the state-of-the-arts on classic VLN benchmarks, with over 17% improvement in success rate. Additionally, our single-stage locomotion policy outperforms previous policy distillation-based methods by a substantial margin. To better simulate the challenges of locomotion navigation in VLN, we introduce a new benchmark, VLN-CE-Isaac, using Isaac Sim. This benchmark considers detailed robotic joint movements and interactions with environments, which prior VLN works have not explored. In our VLN-CE-Isaac experiments, our vision-based policy outperforms the blind policy by a significant margin, showing a 14% improvement in success rate. We also demonstrate that our VLA can be deployed across different robots (Unitree Go2, Unitree H1, Booster T1), each using distinct locomotion skills. Finally, we deploy NaVILA in the real world, exhibiting impressive robustness and achieving an 88% success rate on 25 instructions, including a 75% success rate on complex instructions across diverse scenes.

II. Method

NaVILA integrates high-level visual language understanding with low-level locomotion control (Fig.2). It employs a VLM to process single-view images and generate waypoint instructions in natural language, which a locomotion policy translates into precise joint movements for real-time robot control. The synergy between the VLM's reasoning and the locomotion policy's execution enables NaVILA to generalize across diverse environments. We first describe how we tame VLMs for high-level VLN in Sec.II-A, then outline our robot configuration and locomotion policy in Sec. II-B.

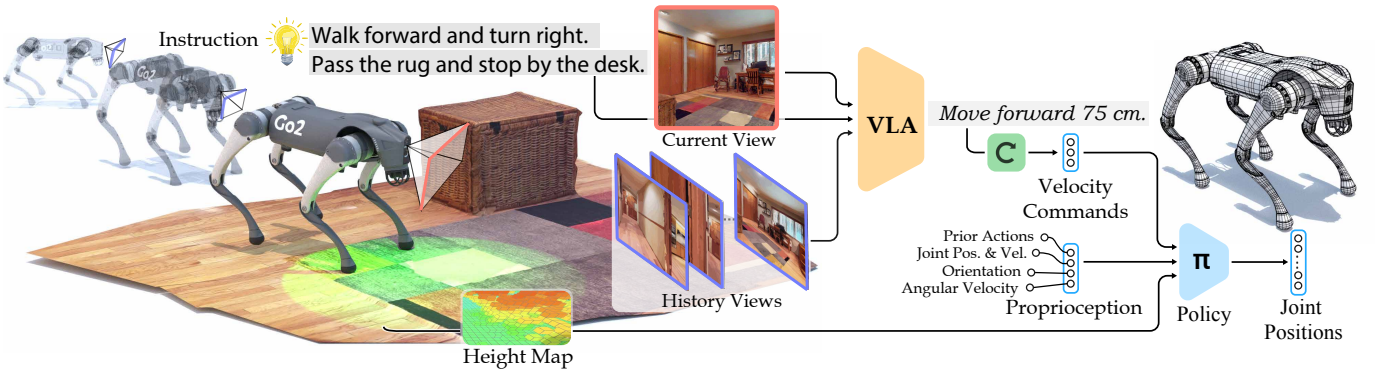


Fig. 2: NaVILA is a two-level framework combining high-level visual language understanding with low-level locomotion control. Our VLA model processes single-view images to produce mid-level actions in natural language, which are then converted into precise joint movements by an advanced low-level locomotion policy. This integration allows for strong generalization and adaptability across different real-world environments, and can operate the robot in real-time.

A. Taming VLMs for Vision Language Navigation

VLN requires processing video inputs as observations. A common approach to handling video inputs in VLMs is through video encoders [12]. However, recent progress in VLMs has largely been driven by the availability of image-text data. While there have been efforts to extend this success to video encoders, the lack of large, high-quality video-text datasets has limited their pre-training. To address this challenge, we opt for image-based vision-language models in our approach. These models exhibit stronger generalization abilities and possess broader knowledge, making them more suitable for tackling the generalization challenges in VLN. Specifically, we built our approach upon VILA [13–19], a family of efficient VLMs for both understanding and generation. VILA's pre-training has proven particularly effective for multi-image reasoning, making it especially suitable for VLN tasks where understanding sequential image relationships is critical.

VILA Preliminary. VILA consists of three main components: a vision encoder, a projector, and an LLM. The vision encoder processes the input images, converting them into a sequence of visual tokens. These tokens are then downsampled and mapped into the language domain via an MLP projector. Afterward, the projected tokens, along with text tokens, are sent to the LLM for auto-regressive generation. When handling videos, VILA uniformly sampled frames at regular intervals. It puts all the frame information before any text. A typical prompt for describing a video might look like "⟨frame3⟩⟨frame6⟩⟨frame9⟩...`Tell me about this video`." Notably, with sequence parallel training [16], VILA can include frames up to 1024. VILA undergoes a 3-stage training process: first, it pre-trains a connector between the frozen LLM and vision backbones using alignment data [20]; then it pre-trains both the connector and the LLM using text-image interleaved corpus [21, 22]; and finally, it fine-tunes all modules (vision encoder, connector, LLM) with instruction tuning data [20, 23].

Navigation Prompts. In vision-language navigation tasks, images from different time steps serve two distinct purposes. The image at time step t represents the current observation, which is crucial for a VLN agent to make immediate decisions (e.g., turning right at an intersection or stopping when the goal is reached). On the other hand, frames before time step t are historical frames that function as a memory bank, helping the agent track overall progress (e.g., remembering the starting location, reasoning about places already visited and planning the next step). Uniformly sampling frames at regular intervals, as done in VILA, is not ideal because it doesn't differentiate between these two types of representations. Therefore, we first extract the most recent frame t as the current observation and then uniformly sample frames from the preceding $t-1$ frames, ensuring the first frame is always included. Additionally, since current and historical observations serve different roles, we distinguish them in our task prompt using textual cues like `a video of historical observations:` for memory frames and `current observation:` for the latest frame. Unlike [12], we avoid introducing additional special tokens that could complicate the LLM's learning process. Instead, we adhere to our design principle of keeping both the input and output of LLM in the language domain to fully leverage the reasoning capabilities of the pre-trained LLM. By integrating these tokens for historical and current observations with the navigation instruction, we construct a navigation task prompt, as shown in Fig. 2.

Learning from Human Videos. Recent studies [24–26] have shown that collecting trajectory-instruction pairs from human videos can enhance navigation capabilities. However, prior work has been limited to discrete navigation settings and has mainly used real videos for pre-training to reduce domain gaps or improve landmark understanding, rather than for directly training navigation models. Extending this approach to continuous settings presents a significant challenge due to the difficulty of obtaining continuous action labels. Recent advances in metric-pose estimation in the wild have now made this feasible, enabling us to extract spatial understanding from human videos and train navigation models directly.

Our data pipeline, shown in Fig. 4, starts with 2K egocentric touring videos from YouTube, which provide a rich source of real-world data to learn robot navigation from human behavior. We process these videos into 20K diverse and representative

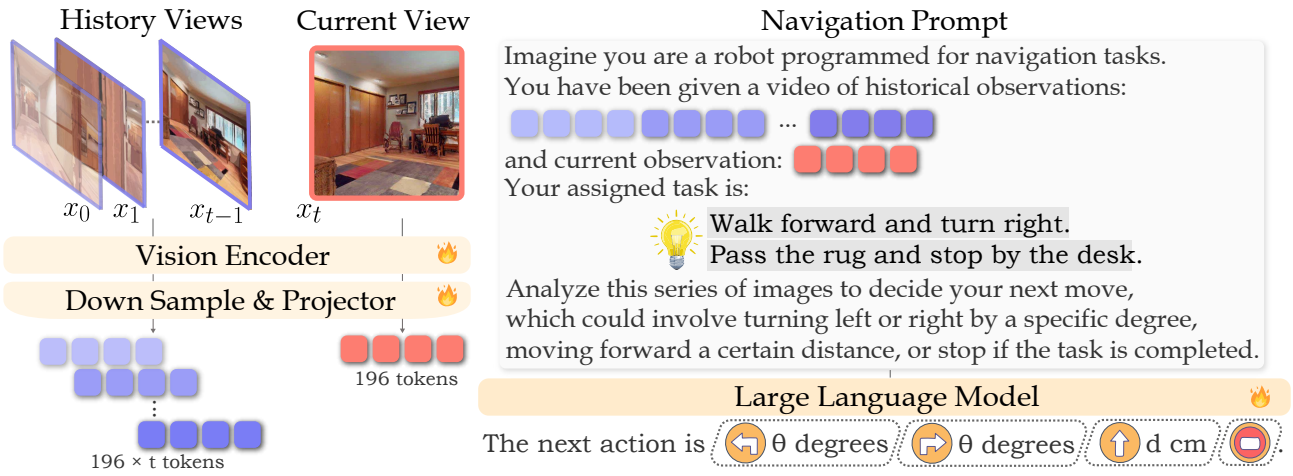


Fig. 3: Overview of our VLA framework. We denote the purple blocks (■) as memory tokens sampled from historical frames, and the red blocks (■) as the current observation tokens. 🔥 denotes trainable parameters. In our experiments, we tested configurations with 8 to 64 frames for t.

trajectories using entropy-based sampling [26]. Next, we estimate camera poses using MASt3R [27] to extract step-by-step actions, and we generate natural language instructions for each trajectory using VLM-based [13] captioning followed by LLM [28] rephrasing. This approach allows us to leverage human demonstrations for continuous navigation, a capability that was previously non-trivial to achieve.

Supervised Fine-tuning Data Blend. Effective Supervised Fine-tuning (SFT) data is crucial for developing a robust vision-language action model. The model should specialize in embodied tasks while avoiding overfitting to specific actions. It should also generalize well to real-world scenarios while retaining broad-world knowledge. Thanks to NaVILA's modular framework, which offers exceptional scalability and adaptability, integrating diverse data sources into our pipeline is straightforward. This flexibility allows us to enhance generalizability for navigation. Our SFT data blend is designed from four perspectives: (1) Navigational data from real videos, (2) Navigational data from simulations, (3) Auxiliary navigational data, and (4) General VQA datasets.

For simulated navigational data, the available VLN datasets in continuous environments are limited, with only R2R-CE [29] and RxR-CE [30] providing sparse path points converted from discrete VLN versions. We leverage both datasets within the Habitat simulator, using a shortest path follower to generate action sequences along the geodesic shortest path. This results in step-wise navigation videos, where each sample comprises a $(t+1)$-frame video and the corresponding oracle action at time step t. To encourage the LLM to generate continuous value labels for distances and angles, we merge consecutive actions (e.g., combining two forward 25 cm steps into a single forward 50 cm step), with a maximum of three consecutive actions. This merging process not only reduces dataset size for more efficient processing but also introduces greater diversity in actions, mitigating overfitting. Additionally, to address label imbalance—particularly the underrepresentation of the stop action—we apply a rebalancing technique for a more even distribution. All navigation-specific data undergo the previously described frame extraction strategy and are paired with navigation task prompts.

To further improve scene understanding and address the limited instructions in R2R-CE and RxR-CE, we incorporate auxiliary navigational datasets. Following [12], we use augmented instructions from EnvDrop [31] and introduce an auxiliary task of navigation trajectory summarization. Given a trajectory video, we sample frames by retaining the first frame and uniformly selecting historical frames, using the annotated instructions as labels. The LLM is then tasked with describing the robot's trajectory based on these frames. To further enhance spatial scene understanding, we integrate the ScanQA [32] dataset, which features real-world 3D scan QA pairs with human-edited questions and free-form answers grounded in 3D objects. For training, we use multi-view RGB images from the raw scans to support this task.

Finally, to maintain the model's general capabilities, we incorporate general VQA datasets from [23, 33, 34]. This comprehensive dataset design ensures that NaVILA can generalize effectively to novel scenes and real-world environments.

Training and Inference Paradigm. Our training process begins with the stage two model of VILA, which has already undergone visual language corpus pre-training. We then apply our SFT data blend to train the entire VLM for one epoch, following standard practices. During this training, all three components—vision encoder, connector, and LLM—are unfrozen. For the inference phase, we implement a regular expression parser [35], to extract action types (such as forward or turn left) and their corresponding arguments (like specific distance or angles) from the LLM output. This method has demonstrated effectiveness in both simulated environments and real-world experiments, where we empirically found that all actions throughout all experiments are successfully matched and mapped.

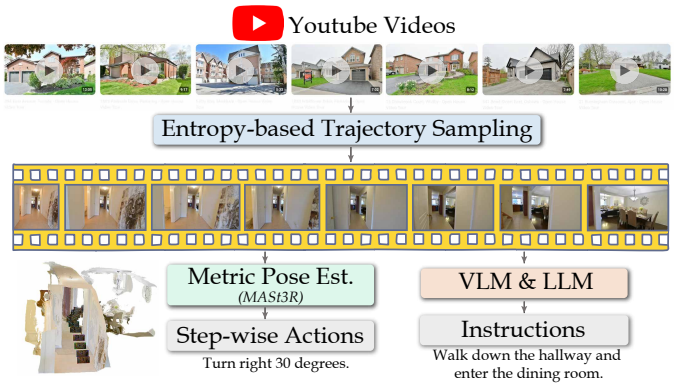


Fig. 4: Data pipeline for transforming human touring videos in the wild into pairwise navigation data within a continuous environment. We begin by processing the videos into meaningful trajectories through entropy-based sampling [26]. Then we extract step-wise actions through metric camera pose estimation [27], and utilize VLM [13] and LLM [28] to generate instructions.

B. Visual Locomotion Policy

In this section, we begin with a brief overview of the Go2 robot dog, the experimental platform used in this work. Next, we describe the development of the end-to-end vision-based control policy, which interprets high-level language navigation commands from the VLM and converts them into precise joint movements. This control policy is trained in the Isaac Sim simulator using Isaac Lab [36] and then directly deployed to the real-world robot.

Go2 Robot. As shown in Fig. 5, the robot is equipped with a LiDAR sensor mounted at the base of its head, broadcasting point clouds at a frequency of 15Hz. The robot features 18 degrees of freedom (DoFs), comprising 6 DoFs for its base and 3 DoFs for each of its four legs. In the policy training process, we left the 6 DoFs on the base unconstrained so that the policy only controls the 12 joint motors on the legs.

Interpreting High-level Commands. As in our formulation, VLM outputs a fixed set of actionable words, such as {move forward, turn left, turn right, stop}, we casts these instructions to fixed command velocities $\{0.5\,\mathrm{m\,s^{-1}}, \frac{\pi}{6}\,\mathrm{rad\,s^{-1}}, -\frac{\pi}{6}\,\mathrm{rad\,s^{-1}}, 0\}$ and execute with corresponding time durations to align with the specific VLM value.

Low-level Action and Observation Space. The action space $\mathbf{a}$ of the control policy is defined as the desired joint position $q^d \in \mathbb{R}^{12}$, which is converted into torque input for the simulator using the stiffness and dampness. We adopt the PPO algorithm [37] to train the policy. During training, the critic observes the privileged environment and generates a value function to update the actor, while the actor only receives sensor data available in the real world. The observation space of the critic $\mathbf{o}^c$ contains the proprioception and velocity command at the current time step t and a privileged terrain height scan around the robot. The proprioceptive data includes robot linear and angular velocity, orientation, joint positions, joint velocities, and the previous action. In the actor's observation space, $\mathbf{o}^a$, linear velocity is excluded, as it is unavailable in the real world, and instead, a history of proprioceptive data is used to infer this information implicitly. The robot perceives the surrounding terrain using a height map from the LiDAR sensor.

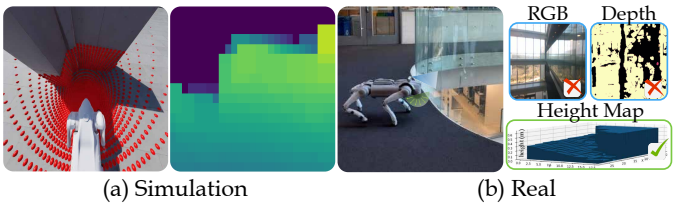


Fig. 5: Height map reconstruction from point cloud. (**a**) Go2 robot follows velocity commands while avoiding obstacles in simulation. Red dots show LiDAR points raycasting from the sensor center to the terrain mesh. The right image shows a preprocessed height map with values clipped to sensor constraints; darker colors indicate higher heights. (**b**) Safe locomotion near glass. The top-down height map detects glass surfaces where depth and RGB images fail.

Incorporating Height Map from LiDAR Point Cloud. Given LiDAR's superior ability to detect transparent objects and robust performance under strong sunlight, we chose the manufacturer-provided LiDAR as the primary sensor for perceiving the robot's surroundings and ensuring safe navigation. The Unitree L1 generates point clouds with a wide field of view of $360° \times 90°$, from which we create a 2.5D height map based on the parameters listed in the Supplementary. For each voxel grid, the lowest value within the range is selected, and a maximum filter is then applied over the last 5 lidar point clouds to smooth the resulting height map.

Training. Different from most existing works [38–41] that utilize the two-stage teacher-student training paradigm, we adopt a single-stage manner to train the locomotion policy. Compared to two-stage training, single-stage RL is more time-efficient as it eliminates the need for policy distillation. Additionally, the policy interacts directly with the environment, allowing it to explore and potentially discover novel strategies. With the support of ray-casting in Isaac Lab, our vision-based RL policy training achieves a high throughput over 60K FPS on an RTX 4090 GPU.

III. Experiments

We conduct experiments to answer the following questions: (1) How does our VLA's performance compare to state-of-the-art methods in VLN-CE benchmarks and general spatial scene understanding tasks? (Sec. III-A) (2) How does the performance of our single-stage visual locomotion policy compare to policy distillation-based approaches? (Sec. III-B) (3) How to evaluate locomotion navigation in simulators, and how effective and flexible is NaVILA in these scenarios? (Sec. III-C) (4) Can NaVILA pipeline be successfully deployed in real robot VLN experiments? (Sec. III-D)

A. High-level VLA Performance

VLN-CE Benchmarks. We evaluate our VLA on the VLN-CE benchmarks, which provide continuous environments for executing navigational actions in reconstructed photorealistic indoor scenes. We focus on the val-unseen split in both R2R (Room-to-Room) and RxR (Room-across-Room) datasets

TABLE I: Comparison with state-of-the-art methods on the Val-Unseen split of R2R-CE [29] and RxR-CE [30]. * indicates methods using the waypoint predictor from Hong et al. [42]. NaVILA outperforms all methods that do not rely on simulator pre-trained waypoint predictors, even when those methods leverage additional inputs such as depth, panoramic views, and odometry.

	Observation				R2R Val-Unseen				RxR Val-Unseen			
	S.RGB	Pano.	Depth	Odo.	NE ↓	OS ↑	SR ↑	SPL ↑	NE ↓	SR ↑	SPL ↑	nDTW ↑
HPN+DN* [43]		✓	✓	✓	6.31	40.0	36.0	34.0	-	-	-	-
CMA* [42]		✓	✓	✓	6.20	52.0	41.0	36.0	8.76	26.5	22.1	47.0
VLN⟳BERT* [42]		✓	✓	✓	5.74	53.0	44.0	39.0	8.98	27.0	22.6	46.7
Sim2Sim* [44]		✓	✓	✓	6.07	52.0	43.0	36.0	-	-	-	-
GridMM* [45]		✓	✓	✓	5.11	61.0	49.0	41.0	-	-	-	-
Ego^2-Map* [46]		✓	✓	✓	5.54	56.0	47.0	41.0	-	-	-	-
DreamWalker* [47]		✓	✓	✓	5.53	59.0	49.0	44.0	-	-	-	-
Reborn* [48]		✓	✓	✓	5.40	57.0	50.0	46.0	5.98	48.6	42.0	63.3
ETPNav* [49]		✓	✓	✓	4.71	65.0	57.0	49.0	5.64	54.7	44.8	61.9
HNR* [50]		✓	✓	✓	4.42	67.0	61.0	51.0	5.50	56.3	46.7	63.5
BEVBert* [51]		✓	✓	✓	4.57	67.0	59.0	50.0	4.00	68.5	-	69.6
HAMT+ScaleVLN* [52]		✓	✓	✓	4.80	-	55.0	51.0	-	-	-	-
AG-CMTP [53]		✓	✓	✓	7.90	39.0	23.0	19.0	-	-	-	-
R2R-CMTP [53]		✓	✓	✓	7.90	38.0	26.0	22.0	-	-	-	-
LAW [54]	✓		✓	✓	6.83	44.0	35.0	31.0	10.90	8.0	8.0	38.0
CM2 [55]	✓		✓	✓	7.02	41.0	34.0	27.0	-	-	-	-
WS-MGMap [56]	✓		✓	✓	6.28	47.0	38.0	34.0	-	-	-	-
AO-Planner [57]		✓	✓		5.55	59.0	47.0	33.0	7.06	43.3	30.5	50.1
Seq2Seq [58]	✓		✓		7.77	37.0	25.0	22.0	12.10	13.9	11.9	30.8
CMA [58]	✓		✓		7.37	40.0	32.0	30.0	-	-	-	-
RGB-Seq2Seq [58]	✓				10.10	8.0	0.0	0.0	-	-	-	-
RGB-CMA [58]	✓				9.55	10.0	5.0	4.0	-	-	-	-
NaVid [12]	✓				5.47	49.0	37.0	35.0	-	-	-	-
NaVILA	✓				**5.22**	**62.5**	**54.0**	**49.0**	**6.77**	**49.3**	**44.0**	**58.8**

TABLE II: Cross-dataset performance on the RxR-CE [30] Val-Unseen split. All results are obtained without training on the RxR-CE training set. NaVILA significantly outperforms NaVid [12], the current single-view state-of-the-art.

	Observation			RxR Val-Unseen			
	S.RGB	Depth	Odo.	NE ↓	OS ↑	SR ↑	SPL ↑
LAW [54]	✓	✓	✓	10.87	21.0	8.0	8.0
CM2 [55]	✓	✓	✓	8.98	25.3	14.4	9.2
WS-MGMap [56]	✓	✓	✓	9.83	29.8	15.0	12.1
Seq2Seq [58]	✓	✓		11.8	5.02	3.51	3.43
CMA [58]	✓	✓		11.7	10.7	4.41	2.47
RGB-Seq2Seq [12]	✓			11.2	12.2	0.0	0.0
RGB-CMA [12]	✓			9.55	14.8	0.0	0.0
A^{2}NAV [59]	✓			-	-	16.8	6.3
NaVid [12]	✓			**8.41**	34.5	23.8	21.2
NaVILA	✓			8.78	**46.8**	**34.3**	**28.2**

within VLN-CE, as these are the two most recognized benchmarks in VLN. We employ the following widely used evaluation metrics for VLN tasks: Navigation Error (NE), Oracle Success Rate (OS), Success Rate (SR), Success-weighted Path Length (SPL), and normalize dynamic time wrapping (nDTW). We show results in Table I, where NaVILA significantly surpasses all baselines that do not rely on simulator pre-trained waypoint predictors in both benchmarks using a single model. Notably, this also marks the first time a VLN agent, trained solely on single-view RGB input, achieves comparable or superior results to models that use panoramic views, odometry, or simulator-pretrained waypoint predictors. This suggests that NaVILA's strong generalization capabilities can effectively compensate for the limited observations in RGB views or sensors.

To evaluate the cross-dataset performance, we follow [12] by training NaVILA exclusively on R2R samples, while leaving out the RxR training set. We then evaluate its zero-shot performance on the RxR Val-Unseen split. As shown in Table II, our method significantly outperforms NaVid, the current state-of-the-art model, with a substantial 10% improvement in SR.

Spatial Scene Understanding Benchmarks. As a general navigation agent, robust spatial scene understanding (e.g., object localization, referring, and spatial reasoning) is crucial. To evaluate NaVILA's capabilities in scene understanding, we conduct evaluations on the ScanQA Validation benchmark, a widely used dataset for 3D Question Answering. ScanQA is based on real-world scans, and we use multi-view images from these scans as input to query NaVILA for answers. As shown in Table III, NaVILA significantly outperforms the previous state-of-the-art model, NaviLLM [60], by a substantial margin (20 points higher on the CIDEr score). Moreover, when using 64 frames, NaVILA's performance demonstrates superior performance compared to state-of-the-art 3D-based large multi-modal models [61, 62]. This is particularly noteworthy as these other models require either 3D scans or RGBD data with camera poses as inputs, while our method achieves better results with less observation.

B. Low-level RL Policy Performance

To highlight the advantages of our RL policy over policy distillation-based approaches, we compared it to Regularized Online Adaptation (ROA) [68]. In ROA training, the model first learns a privileged encoder that processes height scan points and other privileged observations. This privileged encoder then supervises an adaptation encoder, which takes the same 2.5D heightmap as our low-level policy as input. We evaluated both approaches using three metrics: linear velocity

TABLE III: Evaluation of spatial scene understanding performance on the ScanQA dataset [32] Validation split. NaVILA outperforms current state-of-the-art VLA models and demonstrates superior performance to other 3D LMMs that require additional input, such as depth or camera pose. Note that $*$ indicates 3D LMMs that require task-specific fine-tuning on the ScanQA dataset.

	ScanQA Validation				
	Bleu-4 ↑	Rouge ↑	Cider ↑	Meteor ↑	EM ↑
Task-specific Specialist					
VoteNet+MCAN [63]	6.2	29.8	54.7	11.4	17.3
ScanRefer+MCAN [63]	7.9	30.0	55.4	11.5	18.6
ScanQA [32]	10.1	33.3	64.9	13.1	21.0
3D-VisTA [64]	10.4	35.7	69.6	13.9	22.4
3D Large Multi-modal Models					
3D-LLM$_{(flamingo)}$* [65]	7.2	32.3	59.2	12.2	20.4
3D-LLM$_{(BLIP2-flant5)}$* [65]	12.0	35.7	69.4	14.5	20.5
LL3DA* [66]	13.5	37.3	76.8	15.9	-
Chat-3Dv2* [67]	14.0	-	87.6	-	-
Scene-LLM* [61]	12.0	40.0	80.0	16.6	27.2
LEO [62]	13.2	49.2	101.4	20.0	24.5
2D Vision-Langauge-Action Models					
NaviLLM [60]	12.0	38.4	75.9	15.4	23.0
NaVILA (8 frames)	14.8	46.4	95.1	18.7	27.0
NaVILA (64 frames)	**16.9**	**49.3**	**102.7**	**20.1**	**28.6**

error, angular velocity error, and collision rate. The first two metrics assess how accurately the policy follows velocity commands, while the third measures the model's obstacle avoidance capabilities. As shown in Table V, our low-level policy outperforms ROA in all three metrics, particularly achieving a significantly lower collision rate, demonstrating the effectiveness of our training approach.

C. Legged Robot Navigation Performance in Simulation

High-fidelity VLN-CE-Isaac Benchmark. Currently, there are no VLN-CE benchmarks tailored specifically for legged robots. Existing benchmarks [29, 30] for vision-language navigation are based on the Habitat [69] simulator, which focuses on high-level planning without addressing precise low-level robotic control. For instance, agents in Habitat can navigate through narrow gaps, such as a 10 cm space between two sofas, which is impractical for legged robots like quadrupeds or humanoids. To overcome this limitation, we introduce a new benchmark VLN-CE-Isaac built on Isaac Sim. Isaac Sim's high-fidelity simulation captures detailed robotic joint movements and interactions with the environment, enabling comprehensive evaluations of the entire navigation pipeline, from high-level planning to precise robotic execution. We incorporate the same scenes from R2R, with robots deployed in the environment, as shown in Fig. 6. From the 1,839 trajectories in the R2R Val-Unseen split, we select 1,077 traversable trajectories with high-quality meshes to ensure realistic navigation scenarios. For consistency, we evaluate performance using the same metrics as prior work.

Notably, VLN-CE-Isaac is compatible with a variety of robotic platforms. To demonstrate this flexibility, we test our NaVILA model on a Unitree Go2 robot and also a Unitree H1 robot within the benchmark. To highlight the effectiveness of the vision-based policy, we compare it against a proprioception-only (*blind*) policy. As shown in Table IV,

Fig. 6: VLN-CE-Isaac Benchmark visualization.

TABLE IV: VLN-CE-Isaac evaluation results.

	Low-level Observation			VLN-CE-Isaac			
	Proprio.	LiDAR	Height Scan	NE ↓	OS ↑	SR ↑	SPL ↑
Oracle				*5.25*	*59.8*	*51.3*	*46.9*
Unitree Go2							
NaVILA-Blind	✓			6.03	49.0	36.2	33.3
NaVILA-Vision	✓	✓		**5.49**	**58.7**	**50.2**	**45.5**
Unitree H1							
NaVILA-Blind	✓			7.67	33.3	24.4	21.0
NaVILA-Vision	✓		✓	**5.86**	**54.6**	**45.3**	**40.3**

TABLE V: Low level policy performance.

	Linear Vel. Error ↓	Angular Vel. Error ↓	Collision Rate ↓
ROA$_{(w/BCLoss)}$ [68]	0.189	0.152	3.25
ROA [68]	0.161	0.152	3.09
NaVILA	**0.066**	**0.113**	**0.81**

the vision-based policy outperforms the blind policy by 14% in Success Rate in Go2 settings and 21% in H1 settings, owing to its superior obstacle avoidance capability. We also compare NaVILAs with a baseline using Oracle's low-level policy (assuming perfect command execution without realistic physics). Results show a 15% lower success rate on the Go2 setup and a 27% lower success rate on the H1 setup when Oracle policy is not presented. These performance gaps highlight the increased challenges and realism introduced by our benchmark. Additionally, we also observe that the success rate of NaVILA on the H1 robot is significantly lower than on the Go2, which is expected due to the larger size of the humanoid robot.

D. Real World Evaluation

We then conduct experiments in the real world, using 25 instructions, each repeated three times, covering both simple and complex tasks across three types of environments: `Workspace`, `Home`, and `Outdoor` open environments. Simple instructions consist of one or two navigation commands, where the robot does not need to navigate between rooms (e.g., "Go to the chair and stop"). In contrast, complex instructions involve three or more commands, requiring the robot to traverse multiple rooms or landmarks (e.g., "Walk out of the room, turn right, enter the room in front of you, and stop at the table"). We use standard metrics (SR and NE) and compare NaVILA against GPT-4o, a state-of-the-art VLM known for its strong generalizability. As shown in Table VI, NaVILA significantly outperforms GPT-4o in both SR and NE. We also ablate the effectiveness of adding human videos, and the

TABLE VI: Real-world experiments on quadruped (Unitree Go2) and humanoid (Booster T1) conducted in different environments (`Workspace`, `Home`, and `Outdoor`). Simple and Complex refer to simple and complex instruction-following tasks, respectively. Note that † indicates models trained without human touring videos.

	`Workspace`				`Home`				`Outdoor`			
	Simple		Complex		Simple		Complex		Simple		Complex	
	NE↓	SR↑	NE↓	SR↑	NE↓	SR↑	NE↓	SR↑	NE↓	SR↑	NE↓	SR↑
Unitree Go2												
GPT-4o [28]	2.01	0.67	2.38	0.33	1.49	0.53	3.00	0.00	-	0.67	-	0.50
NaVILA †	2.00	0.60	1.81	0.73	2.17	0.47	2.32	0.40	-	0.00	-	0.00
NaVILA	**1.29**	**1.00**	**1.76**	**0.80**	**1.15**	**1.00**	**1.76**	**0.67**	-	**1.00**	-	**0.83**
Booster T1												
GPT-4o [28]	1.53	0.67	2.78	0.13	-	-	-	-	-	0.44	-	-
NaVILA †	2.36	0.40	2.16	0.62	-	-	-	-	-	0.22	-	-
NaVILA	**1.18**	**0.93**	**1.91**	**0.67**	-	-	-	-	-	**0.89**	-	-

results show that with the help of human videos, the model can generalize better to outdoor scenes and achieve higher success rates across all environments. To demonstrate the flexibility of our two-level approach, we also evaluated it on a Booster Dynamics T1 humanoid robot, using the same VLA model without any retraining. Despite variations such as changes in camera height and camera view angle, NaVILA consistently outperformes the baselines, highlighting the strong generalization capabilities of our model. Our qualitative results are presented in Fig.1 and Fig.7. In Fig. 7, we demonstrate integration with speech recognition, enabling voice-controlled navigation through our framework. These results highlight the effectiveness of NaVILA in bridging the gap between vision-language understanding and real-world navigation tasks.

IV. Related Work

Visual Navigation. Visual navigation has been a long-standing research topic in robotics [71–74]. Classical methods rely on pre-computed [75] or geometric maps built with depth sensors [76] or monocular cameras while localizing the robot (SLAM) [77, 78]. More recently, learning-based approaches using Imitation Learning [79, 80] and Reinforcement Learning [81, 82] have demonstrated strong performance and expanded applications to Vision-Language Navigation.

Vision-Language Navigation. Vision-Language Navigation (VLN) is a fundamental challenge in embodied AI, where agents navigate complex environments using visual cues and natural language instructions. The field has evolved significantly over time. Early research [1, 30, 83] focused on discrete navigation in simulated environments like MP3D [84], where agents teleport between predefined nodes on a navigation graph [31, 85–91]. As foundation models advanced, many VLN systems improved dramatically by leveraging large-scale pre-trained models [25, 92] and pre-training techniques [24, 52, 93], approaching human-level performance in this setting. However, this setup emphasized high-level decision-making while neglecting the challenges of underlying motion control. Recently, research [12, 54–57] has shifted towards continuous environments (VLN-CE [58]) using simulators like Habitat [69]. This introduces greater complexity, as agents must perform mid-level actions such as moving forward or rotating, rather than teleporting between nodes. To bridge the gap between discrete and continuous navigation, some approaches [44, 49, 51, 94] use simulator pre-trained waypoint models [42, 43] that predict candidate positions around the agent and have shown significant performance gains. However, they often struggle to generalize due to their reliance on simulator-specific data. Additionally, the candidate positions predicted by these models only cover nearby locations and do not account for low-level motion planning or obstacle avoidance. In this paper, we aim to advance VLN toward real-world robotics applications, particularly for challenging-legged robots. NaVILA handles both high-level decision-making and generates low-level actions to control the robot's full motion. Additionally, we introduce a new VLN benchmark built on Isaac Sim, offering a more realistic simulation environment, which we believe will benefit future work in VLN.

Robot Foundation Models. Robot foundation models aim to provide a unified framework that processes inputs from various modalities (e.g., vision and language) and directly outputs actions to enable robots to perform complex tasks. Existing works [7, 8, 95] trained on large-scale robotic datasets to get general robot policies, but mainly focusing on manipulation tasks. Doshi et al. [96] and Yang et al. [97] proposed end-to-end visual-language cross-embodiment models for different robotic tasks. Recently, several foundational navigation models have been proposed [98–100]. However, they mainly focus on goal navigation with either short language descriptions or a target image as input. As for legged robots, Ding et al. [101] proposed a unified model to leverage vision and language inputs and generate executable low-level actions. Another line of work [102, 103] focuses on training specialized policies as a skill bank to handle specific actions, with either a VLM or LLM serving as the controller to decide which skill to execute. Similarly, these methods cannot perform instruction-following tasks, as they struggle to comprehend complex instructions that are essential for general navigation. To address this, we propose a VLA model specifically designed for general vision language navigation tasks.

Legged Robot Locomotion Learning. Legged robot locomotion learning focuses on enabling robots to traverse various terrains. Previous works [104, 105] rely solely on robot's proprioceptive information struggle in scenarios like obstacle avoidance. Other end-to-end vision-based approaches [106–109] are vulnerable to extreme environmental conditions, such as intense sunlight, due to the limitations of sensors. Lee et al. [38] incorporate LiDAR sensors in addition to depth cameras to improve terrain sensing, but rely on time-inefficient two-stage training. Additionally, during training, Miki et al. [39] query predefined terrain heights to construct a height map and then rely on an external tool[110] for height map generation during deployment, resulting in discrepancies between training and deployment. To overcome these limitations, we propose a single-stage RL framework that integrates LiDAR sensing inputs during training, allowing the robot to directly learn from interacting with the environments for better efficiency and robustness in complex scenarios.

Fig. 7: Qualitative results from the real-world deployment of NaVILA. (a) We integrate speech recognition [70] into NaVILA, allowing a human to control the robot using voice commands that begin with "`Hey Robot!`". (b) The robot successfully handles long-horizon navigation tasks. Given a lengthy instruction, it moves through different areas of the house and stops at the specified goal. (c), (d), and (e) The robot demonstrates its ability to navigate through obstacles, traverse challenging terrains, and climb up and down stairs.

V. Conclusion and Limitations

We introduce NaVILA, a two-level framework that unifies VLAs with locomotion skills for generic navigation. NaVILA generates high-level language commands while a real-time locomotion policy handles obstacle avoidance, enhancing robustness across robots. This design preserves reasoning, prevents overfitting, and enables direct learning from human videos for better generalization. NaVILA achieves a 17% gain on classic VLN benchmarks, outperforms distillation-based low-level policies, surpasses vision-blind policies on VLN-CE-Isaac1K, and demonstrates strong real-world performance across diverse environments and legged robots.

Limitations. While NaVILA demonstrates strong performance, it fails in some real-world cases (see Appx. Sec. E). Enhancing generalizability and spatial understanding through larger-scale training in realistic simulations could help. Additionally, image-based vision-language models are computationally intensive. Advances in long-context LLMs may alleviate this by enabling more efficient sequence processing.

References

[1] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018. 2, 8

[2] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *CVPR*, 2019.

[3] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *ICLR*, 2020.

[4] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *NeurIPS*, 2020.

[5] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *CVPR*, 2020.

[6] Ram Ramrakhya, Eric Undersander, Dhruv Batra, and Abhishek Das. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *CVPR*, 2022. 2

[7] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint*, 2023. 2, 8

[8] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint*, 2024. 8

[9] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. In *ICRA*, 2024. 2

[10] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *CVPR*, 2024. 2

[11] An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision-language models. In *NeurIPS*, 2024. 2

[12] Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and Wang He. Navid: Video-based vlm plans the next step for vision-and-language navigation. In *RSS*, 2024. 3, 4, 6, 8, 1

[13] Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. Vila: On pre-training for visual language models. In *CVPR*, 2024. 3, 4, 5

[14] Yecheng Wu, Zhuoyang Zhang, Junyu Chen, Haotian Tang, Dacheng Li, Yunhao Fang, Ligeng Zhu, Enze Xie, Hongxu Yin, Li Yi, et al. Vila-u: a unified foundation model integrating visual understanding and generation. *arXiv preprint*, 2024.

[15] Yunhao Fang, Ligeng Zhu, Yao Lu, Yan Wang, Pavlo Molchanov, Jan Kautz, Jang Hyun Cho, Marco Pavone, Song Han, and Hongxu Yin. Vila ^{2}: Vila augmented vila. *arXiv preprint*, 2024.

[16] Fuzhao Xue, Yukang Chen, Dacheng Li, Qinghao Hu, Ligeng Zhu, Xiuyu Li, Yunhao Fang, Haotian Tang, Shang Yang, Zhijian Liu, et al. Longvila: Scaling long-context visual language models for long videos. *arXiv preprint*, 2024. 3

[17] Hanrong Ye, De-An Huang, Yao Lu, Zhiding Yu, Wei Ping, Andrew Tao, Jan Kautz, Song Han, Dan Xu, Pavlo Molchanov, et al. X-vila: Cross-modality alignment for large language model. *arXiv preprint*, 2024.

[18] De-An Huang, Shijia Liao, Subhashree Radhakrishnan, Hongxu Yin, Pavlo Molchanov, Zhiding Yu, and Jan Kautz. Lita: Language instructed temporal-localization assistant. In *ECCV*, 2024.

[19] Zhijian Liu, Ligeng Zhu, Baifeng Shi, Zhuoyang Zhang, Yuming Lou, Shang Yang, Haocheng Xi, Shiyi Cao, Yuxian Gu, Dacheng Li, et al. Nvila: Efficient frontier visual language models. *arXiv preprint*, 2024. 3

[20] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 3

[21] Minwoo Byeon, Beomhee Park, Haecheon Kim, Sungjun Lee, Woonhyuk Baek, and Saehoon Kim. Coyo-700m: Image-text pair dataset. https://github.com/kakaobrain/coyo-dataset, 2022. 3

[22] Wanrong Zhu, Jack Hessel, Anas Awadalla, Samir Yitzhak Gadre, Jesse Dodge, Alex Fang, Youngjae Yu, Ludwig Schmidt, William Yang Wang, and Yejin Choi. Multimodal c4: An open, billion-scale corpus of images interleaved with text. In *NeurIPS*, 2024. 3

[23] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *CVPR*, 2024. 3, 4

[24] Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. Airbert: In-domain pretraining for vision-and-language navigation. In *ICCV*, 2021. 3, 8

[25] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *ECCV*, 2020. 8

[26] Kunyang Lin, Peihao Chen, Diwei Huang, Thomas H Li, Mingkui Tan, and Chuang Gan. Learning vision-and-language navigation from youtube videos. In *ICCV*,

2023. 3, 4, 5

[27] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *ECCV*. Springer, 2024. 4, 5

[28] OpenAI. Hello gpt-4o, 2024. URL https://openai.com/index/hello-gpt-4o/. 4, 5, 8

[29] Jacob Krantz, Erik Wijmans, Arjun Majundar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision and language navigation in continuous environments. In *ECCV*, 2020. 4, 6, 7, 1

[30] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *EMNLP*, 2020. 4, 6, 7, 8

[31] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*, 2019. 4, 8

[32] Daichi Azuma, Taiki Miyanishi, Shuhei Kurita, and Motoaki Kawanabe. Scanqa: 3d question answering for spatial scene understanding. In *CVPR*, 2022. 4, 7

[33] Lin Chen, Jisong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. Sharegpt4v: Improving large multi-modal models with better captions. In *ECCV*, 2024. 4

[34] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *ACL*, 2024. 4

[35] Steven M Kearns. Extending regular expressions with context operators and parse extraction. *Software: Practice and Experience*, 1991. 4

[36] Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Ritvik Singh, Yunrong Guo, Hammad Mazhar, Ajay Mandlekar, Buck Babich, Gavriel State, Marco Hutter, and Animesh Garg. Orbit: A unified simulation framework for interactive robot learning environments. In *RAL*, 2023. 5

[37] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint*, 2017. 5

[38] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 2020. 5, 8

[39] Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 2022. 8

[40] Gabriel B Margolis, Ge Yang, Kartik Paigwar, Tao Chen, and Pulkit Agrawal. Rapid locomotion via reinforcement learning. In *RSS*, 2022.

[41] Joonho Lee, Marko Bjelonic, Alexander Reske, Lorenz Wellhausen, Takahiro Miki, and Marco Hutter. Learning robust autonomous navigation and locomotion for wheeled-legged robots. *Science Robotics*, 2024. 5

[42] Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *CVPR*, 2022. 6, 8

[43] Jacob Krantz, Aaron Gokaslan, Dhruv Batra, Stefan Lee, and Oleksandr Maksymets. Waypoint models for instruction-guided navigation in continuous environments. In *CVPR*, 2021. 6, 8

[44] Jacob Krantz and Stefan Lee. Sim-2-sim transfer for vision-and-language navigation in continuous environments. In *ECCV*, 2022. 6, 8

[45] Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, and Shuqiang Jiang. Gridmm: Grid memory map for vision-and-language navigation. In *ICCV*, 2023. 6

[46] Yicong Hong, Yang Zhou, Ruiyi Zhang, Franck Dernoncourt, Trung Bui, Stephen Gould, and Hao Tan. Learning navigational visual representations with semantic map supervision. In *ICCV*, 2023. 6

[47] Hanqing Wang, Wei Liang, Luc Van Gool, and Wenguan Wang. Dreamwalker: Mental planning for continuous vision-language navigation. In *ICCV*, 2023. 6

[48] Dong An, Zun Wang, Yangguang Li, Yi Wang, Yicong Hong, Yan Huang, Liang Wang, and Jing Shao. 1st place solutions for rxr-habitat vision-and-language navigation competition. In *CVPRW*, 2022. 6

[49] Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. *IEEE TPAMI*, 2024. 6, 8

[50] Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, Junjie Hu, Ming Jiang, and Shuqiang Jiang. Lookahead exploration with neural radiance representation for continuous vision-language navigation. In *CVPR*, 2024. 6

[51] Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. Bevbert: Multimodal map pre-training for language-guided navigation. In *ICCV*, 2023. 6, 8

[52] Zun Wang, Jialu Li, Yicong Hong, Yi Wang, Qi Wu, Mohit Bansal, Stephen Gould, Hao Tan, and Yu Qiao. Scaling data generation in vision-and-language navigation. In *ICCV*, 2023. 6, 8

[53] Kevin Chen, Junshen K Chen, Jo Chuang, Marynel Vázquez, and Silvio Savarese. Topological planning with transformers for vision-and-language navigation. In *CVPR*, 2021. 6

[54] Sonia Raychaudhuri, Saim Wani, Shivansh Patel, Unnat Jain, and Angel Chang. Language-aligned waypoint (law) supervision for vision-and-language navigation in continuous environments. In *EMNLP*, 2021. 6, 8

[55] Georgios Georgakis, Karl Schmeckpeper, Karan Wanchoo, Soham Dan, Eleni Miltsakaki, Dan Roth, and Kostas Daniilidis. Cross-modal map learning for vision and language navigation. In *CVPR*, 2022. 6

[56] Peihao Chen, Dongyu Ji, Kunyang Lin, Runhao Zeng, Thomas Li, Mingkui Tan, and Chuang Gan. Weakly-supervised multi-granularity map learning for vision-

and-language navigation. In *NeurIPS*, 2022. 6

[57] Jiaqi Chen, Bingqian Lin, Xinmin Liu, Xiaodan Liang, and Kwan-Yee K Wong. Affordances-oriented planning using foundation models for continuous vision-language navigation. *arXiv preprint*, 2024. 6, 8

[58] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *ECCV*, 2020. 6, 8

[59] Peihao Chen, Xinyu Sun, Hongyan Zhi, Runhao Zeng, Thomas H. Li, Gaowen Liu, Mingkui Tan, and Chuang Gan. A^2nav: Action-aware zero-shot robot navigation by exploiting vision-and-language ability of foundation models. *arXiv preprint*, 2023. 6

[60] Duo Zheng, Shijia Huang, Lin Zhao, Yiwu Zhong, and Liwei Wang. Towards learning a generalist model for embodied navigation. In *CVPR*, 2024. 6, 7

[61] Rao Fu, Jingyu Liu, Xilun Chen, Yixin Nie, and Wenhan Xiong. Scene-llm: Extending language model for 3d visual understanding and reasoning. *arXiv preprint*, 2024. 6, 7

[62] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. In *ICML*, 2024. 6, 7

[63] Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. Deep modular co-attention networks for visual question answering. In *CVPR*, 2019. 7

[64] Ziyu Zhu, Xiaojian Ma, Yixin Chen, Zhidong Deng, Siyuan Huang, and Qing Li. 3d-vista: Pre-trained transformer for 3d vision and text alignment. In *ICCV*, 2023. 7

[65] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. In *NeurIPS*, 2023. 7

[66] Sijin Chen, Xin Chen, Chi Zhang, Mingsheng Li, Gang Yu, Hao Fei, Hongyuan Zhu, Jiayuan Fan, and Tao Chen. Ll3da: Visual interactive instruction tuning for omni-3d understanding reasoning and planning. In *CVPR*, 2024. 7

[67] Haifeng Huang, Zehan Wang, Rongjie Huang, Luping Liu, Xize Cheng, Yang Zhao, Tao Jin, and Zhou Zhao. Chat-scene: Bridging 3d scene and large language models with object identifiers. In *NeurIPS*, 2024. 7

[68] Zipeng Fu, Xuxin Cheng, and Deepak Pathak. Deep whole-body control: Learning a unified policy for manipulation and locomotion. In *CoRL*, 2022. 6, 7

[69] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *ICCV*, 2019. 7, 8

[70] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *ICML*, 2023. 9

[71] Hans Peter Moravec. *Obstacle avoidance and navigation in the real world by a seeing robot rover*. Stanford University, 1980. 8

[72] Alberto Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal on Robotics and Automation*, 1987.

[73] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaert. Robust monte carlo localization for mobile robots. *Artificial intelligence*, 2001.

[74] Theophile Gervet, Soumith Chintala, Dhruv Batra, Jitendra Malik, and Devendra Singh Chaplot. Navigating to objects in the real world. *Science Robotics*, 2023. 8

[75] Sebastian Thrun, Maren Bennewitz, Wolfram Burgard, Armin B Cremers, Frank Dellaert, Dieter Fox, Dirk Hahnel, Charles Rosenberg, Nicholas Roy, Jamieson Schulte, et al. Minerva: A second-generation museum tour-guide robot. In *ICRA*, 1999. 8

[76] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *International Symposium on Mixed and Augmented Reality*, 2011. 8

[77] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE TPAMI*, 2007. 8

[78] Eagle S Jones and Stefano Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *The International Journal of Robotics Research*, 2011. 8

[79] Devendra Singh Chaplot, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, and Ruslan Salakhutdinov. Gated-attention architectures for task-oriented language grounding. In *AAAI*, 2018. 8

[80] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *ICRA*, 2018. 8

[81] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 2015. 8

[82] TP Lillicrap. Continuous control with deep reinforcement learning. *arXiv preprint*, 2015. 8

[83] Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. In *CVPR*, 2020. 8

[84] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *3DV*, 2017. 8

[85] Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *NeurIPS*, 2018. 8

[86] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *ICLR*, 2019.

[87] Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *CVPR*, 2019.

[88] Yicong Hong, Cristian Rodriguez, Yuankai Qi, Qi Wu, and Stephen Gould. Language and visual entity relationship graph for agent navigation. In *NeurIPS*, 2020.

[89] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. In *NeurIPS*, 2021.

[90] Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee Wong. Mapgpt: Map-guided prompting with adaptive path planning for vision-and-language navigation. In *ACL*, 2024.

[91] Gengze Zhou, Yicong Hong, Zun Wang, Xin Eric Wang, and Qi Wu. Navgpt-2: Unleashing navigational reasoning capability for large vision-language models. In *ECCV*, 2024. 8

[92] Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Celikyilmaz, Jianfeng Gao, Noah Smith, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. In *EMNLP*, 2019. 8

[93] Aishwarya Kamath, Peter Anderson, Su Wang, Jing Yu Koh, Alexander Ku, Austin Waters, Yinfei Yang, Jason Baldridge, and Zarana Parekh. A new path: Scaling vision-and-language navigation with synthetic instructions and imitation learning. In *CVPR*, 2023. 8

[94] Muhammad Zubair Irshad, Chih-Yao Ma, and Zsolt Kira. Hierarchical cross-modal agent for robotics vision-and-language navigation. In *ICRA*, 2021. 8

[95] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. In *RSS*, 2024. 8

[96] Ria Doshi, Homer Rich Walke, Oier Mees, Sudeep Dasari, and Sergey Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. In *CoRL*, 2024. 8

[97] Jonathan Yang, Catherine Glossop, Arjun Bhorkar, Dhruv Shah, Quan Vuong, Chelsea Finn, Dorsa Sadigh, and Sergey Levine. Pushing the limits of cross-embodiment learning for manipulation and navigation. *arXiv preprint*, 2024. 8

[98] Kuo-Hao Zeng, Zichen Zhang, Kiana Ehsani, Rose Hendrix, Jordi Salvador, Alvaro Herrasti, Ross Girshick, Aniruddha Kembhavi, and Luca Weihs. Poliformer: Scaling on-policy rl with transformers results in masterful navigators. In *CoRL*, 2024. 8

[99] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. Gnm: A general navigation model to drive any robot. In *ICRA*, 2023.

[100] Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. Nomad: Goal masked diffusion policies for navigation and exploration. In *ICRA*, 2024. 8

[101] Pengxiang Ding, Han Zhao, Zhitao Wang, Zhenyu Wei, Shangke Lyu, and Donglin Wang. Quar-vla: Vision-language-action model for quadruped robots. In *ECCV*, 2024. 8

[102] Annie S Chen, Alec M Lessing, Andy Tang, Govind Chada, Laura Smith, Sergey Levine, and Chelsea Finn. Commonsense reasoning for legged robot adaptation with vision-language models. *arXiv preprint*, 2024. 8

[103] Yutao Ouyang, Jinhan Li, Yunfei Li, Zhongyu Li, Chao Yu, Koushil Sreenath, and Yi Wu. Long-horizon locomotion and manipulation on a quadrupedal robot with large language models. *arXiv preprint*, 2024. 8

[104] Yikai Wang, Zheyuan Jiang, and Jianyu Chen. Learning robust, agile, natural legged locomotion skills in the wild. In *CoRL*, 2023. 8

[105] Junfeng Long, Zirui Wang, Quanyi Li, Liu Cao, Jiawei Gao, and Jiangmiao Pang. Hybrid internal model: Learning agile legged locomotion with simulated robot response. In *ICLR*, 2024. 8

[106] Simar Kareer, Naoki Yokoyama, Dhruv Batra, Sehoon Ha, and Joanne Truong. Vinl: Visual navigation and locomotion over obstacles. In *ICRA*, 2023. 8

[107] Ruihan Yang, Minghao Zhang, Nicklas Hansen, Huazhe Xu, and Xiaolong Wang. Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers. In *ICLR*, 2022.

[108] Chieko Sarah Imai, Minghao Zhang, Yuchen Zhang, Marcin Kierebiński, Ruihan Yang, Yuzhe Qin, and Xiaolong Wang. Vision-guided quadrupedal locomotion in the wild with multi-modal delay randomization. In *IROS*, 2022.

[109] Ruihan Yang, Ge Yang, and Xiaolong Wang. Neural volumetric memory for visual locomotion control. In *CVPR*, 2023. 8

[110] Takahiro Miki, Lorenz Wellhausen, Ruben Grandia, Fabian Jenelten, Timon Homberger, and Marco Hutter. Elevation mapping for locomotion and navigation using gpu. In *IROS*, 2022. 8

[111] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration. In *MLSys*, 2024. 3

APPENDIX

TABLE OF CONTENTS

A. More Ablation Studies

1) Different Simulation Data Blends: We perform an ablation study to assess the impact of different simulation data blends on training VLA. As shown in Table VII, training navigation data without label rebalancing leads to a significant drop in performance. Additionally, training VLA exclusively on RxR data demonstrates reasonable cross-dataset performance on R2R-CE, supporting our observations in Table II. Lastly, we investigate whether excluding RxR dataset degrades R2R-CE performance. The results suggest that the RxR dataset does not significantly contribute to the R2R-CE performance.

TABLE VII: Results on R2R-CE using different data blends.

	R2R-CE Val Unseen			
	NE ↓	OSR ↑	SR ↑	SPL ↑
NaVILA † (w/o Label Balancing)	7.82	47.5	30.0	25.1
NaVILA † (w/ RxR only)	7.57	40.8	31.5	27.8
NaVILA † (w/o RxR)	6.11	57.0	47.7	42.4
NaVILA †	**5.37**	**57.6**	**49.7**	**45.5**

2) Human Touring Video Data: We perform ablation studies in simulations to assess the effect of using real-world data from YouTube human touring videos. As shown in Table VIII, incorporating this data leads to significant performance gains, with approximately 5% improvements in OS, SR, and SPL metrics. Similarly, results from real-world experiments (Table VI) demonstrate higher success rates and fewer navigation errors. These results validate the effectiveness of our data pipeline and highlight the scalability of NaVILA's framework, which supports easy integration of data from diverse sources.

TABLE VIII: Results on R2R-CE using additional real data from human touring videos.

	R2R-CE Val Unseen			
	NE ↓	OSR ↑	SR ↑	SPL ↑
NaVILA †	5.37	57.6	49.7	45.5
NaVILA	**5.22**	**62.5**	**54.0**	**49.0**

† indicates models trained without human touring videos.

3) Different Memory Sizes: We conduct an ablation study to evaluate the impact of memory size (number of history frames) on the navigation task using R2R-CE benchmark. The results in Table IX show that for R2R-CE, 8 frames are sufficient to cover most instruction horizons, with limited performance gains from increasing the memory size. For real-world experiments, we use an 8-frame memory size due to latency constraints.

TABLE IX: Ablation study on different memory size using R2R-CE [29] Validation Unseen split.

	R2R-CE Val Unseen			
	NE ↓	OSR ↑	SR ↑	SPL ↑
NaVid [12]	5.47	49.0	37.0	35.0
NaVILA † (8 frames)	**5.37**	57.6	49.7	**45.5**
NaVILA † (16 frames)	5.63	55.8	48.6	44.4
NaVILA † (32 frames)	5.74	55.9	49.5	44.1
NaVILA † (64 frames)	5.63	**60.5**	**50.1**	45.4

B. More Qualitative Results

1) VLN-CE-Isaac: Here we show a visualization example highlighting why the Go2 vision policy significantly outperforms the blind policy. As demonstrated in Figure 8, when encountering an obstacle, the VLA, which is not specifically trained for obstacle avoidance, failed to navigate around it effectively. The blind policy, following the VLA's commands without additional sensory input, became stuck at the obstacle. In contrast, the vision-based policy, trained to handle obstacles using LiDAR input, can autonomously avoid dangers even when the high-level VLA model does not detect them.

Blind Policy

Vision Policy

Fig. 8: Comparison between Go2 blind policy and vision policy. The blind policy failed to avoid the obstacles and got stuck. The vision policy detected the obstacle and got around to avoiding it.

2) Real-World: We show more real-world results in Figure 9. NaVILA demonstrates robust performance and exceptional generalization across diverse settings.

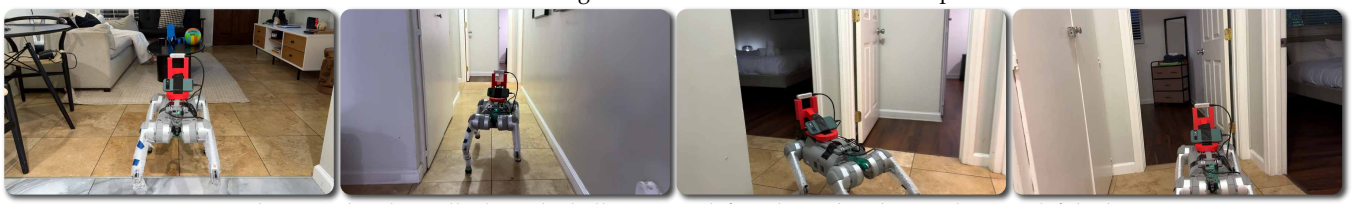
Turn left a little. Go straight forward. Stop when you step on the grass.

Walk forward. Step on the grass and continue going forward.
Stop when you are close to the big bear statue.

Walk to the other end of the room, turn left and find a toy kitchen set.

Move forward into the kitchen. Turn right at the corner, and then walk straight forward. Stop in front of the red bowl.

Walk forward out of the room. Turn right and enter the other room and stop in front of the table.



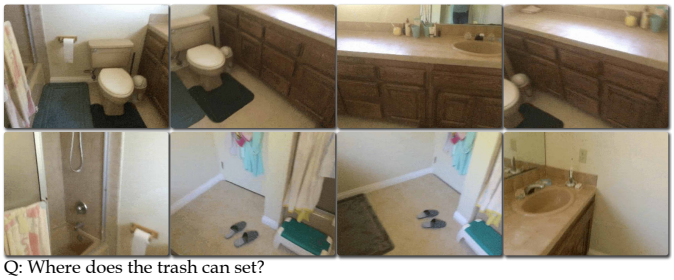
Turn right immediately, walk along the hallway, turn left at the end and enter the most left bedroom.

Walk all the way down, turn left at the intersection and find a bookshelf.

Walk all the way down, turn left at the intersection and find a ball.

Fig. 9: We show more results in diverse environments, such as urban streets, campus sidewalks, courtyards, and different houses. These settings add significant variety and challenges, including challenging terrains, dynamic objects, and different lighting conditions. The results NaVILA achieved represent a significant milestone, showcasing capabilities that have never been demonstrated before.

3) Spatial Scene Understanding: In Figure 10, we show NaVILA's qualitative results on spatial scene understanding using the ScanQA benchmark. Given a sequence of images sampled from a video, NaVILA can ground and locate objects correctly.



Q: Where does the trash can set?
NaVILA: To right of toilet.
Q: What is hanging on the wall between the toilet and the bath tub?
NaVILA: Toilet paper.

Q: What can be seen on the wall on the right of the chair?
NaVILA: Whiteboard.
Q: What color is the mini fridge in the corner of the room?
NaVILA: Black.

Fig. 10: Spatial scene understanding results on ScanQA benchmark.

C. More Implementation Details

1) Video Navigation Trajectory Summarization: We provide the data prompts for our auxiliary task of video navigation trajectory summarization. Following the approach in [12], we construct prompt templates that characterize the LLM as a robot designed for navigation. We process the trajectory videos into history frames, insert the frame tokens into the prompt, and ask the LLM to infer the navigation instructions from the video. This task is designed to enhance the robot's scene understanding and its familiarity with the instruction format.

```
Assume you are a robot designed for navigation. You
are provided with captured image sequences:

<frame3><frame6><frame9>...

Based on this image sequence, please describe the
navigation trajectory of the robot.
```

2) VLA Hyperparameters: Please refer to VILA's paper for details on the hyperparameters used in the first two stages. In the instruction fine-tuning stage, we use a learning rate of $1e^{-4}$ with cosine decay and a warm-up ratio of 0.03. We will release both our training code and data upon paper publication.

3) Locomotion Motion Policy: The reward functions and domain randomization used during Go2 locomotion policy training are listed in Table X and Table XI. The robust policy is trained on flat, rough, slope and obstacle terrains shown in Figure 11. LiDAR and height map settings are detailed in Table XII.

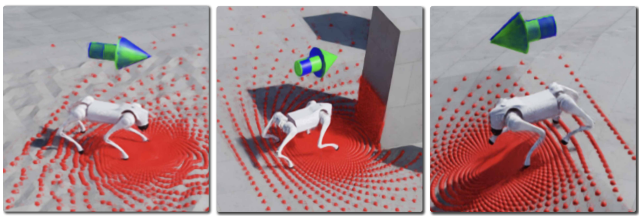


Fig. 11: Random rough, obstacle and slope terrain.

TABLE X: Reward function parameters for training RL policy.

Reward	Expression	Weight
Linear velocity tracking	$\exp(-\lVert v_{xy}^{\text{cmd}} - v_{xy}\rVert_2^2)$	1.5
Angular velocity tracking	$\exp(-\left(\omega_{\text{yaw}}^{\text{cmd}} - \omega_{\text{yaw}}\right)^2)$	1.5
Linear velocity penalty (z)	v_z^2	-2.0
Angular velocity penalty (xy)	$\lVert \boldsymbol{\omega}_{xy}\rVert_2^2$	-0.05
Flat orientation	$\lVert \mathbf{g}\rVert_2^2$	-2.0
Joint accelerations	$\lVert \ddot{\boldsymbol{\theta}}\rVert^2$	-2.5×10^{-7}
Energy	$-\lVert \tau\dot{q}\rVert_2^2$	-2×10^{-5}
Body height	$(h^{\text{target}} - h)^2$	-5.0
Feet slipping	$-\lVert v_{\text{feet}} \cdot \mathbf{1}[F_{\text{feet}} > 1]\rVert_2$	0.05

TABLE XI: Domain randomization parameters for training RL policy.

Parameter	Value
Body Mass	[-3.0, 3.0]
Static Ground Friction	[0.4, 4.0]
Dynamic Ground Friction	[0.4, 4.0]
Motor Strength	[0.9,1.1]
System Delay	$[\Delta_t, \Delta_t]$

TABLE XII: LiDAR and Height Map parameters in simulation.

Parameter	Value
Channels	32
Vertical Range (degrees)	(0, 90)
Horizontal Range (degrees)	(-180, 180)
Horizontal Resolution (degrees)	4
Voxel Size (m)	0.06
X Range (m)	[-0.8, 0.2]
Y Range (m)	[-0.8, 0.8]
Z Range (m)	[0.05, 0.5]

4) VLA Compute Resources: The first two stages of NaVILA are inherited from VILA [13], which is trained on 16 A100 GPU nodes, with each node having 8 GPUs. The training times for each stage of our 8B model are as follows: connector initialization takes 4 hours, visual language pre-training takes 30 hours. The final visual instruction-tuning stage is experimented on 4 A100 GPU nodes, taking 18 hours. During inference time, the VLA model in NaVILA can be served using a single RTX 4090 GPU with roughly 1 FPS.

D. Parameter-efficient Quantization

We explore optimization techniques to improve NaVILA's inference efficiency. Specifically, we apply AWQ [111], a state-of-the-art quantization method for VLMs, to the FP16 NaVILA-8B model. By converting it to the W4A16 format (low-bit weight-only quantization), we achieved significant improvements: memory requirements dropped by half, and processing speed improved by about 40%. Most importantly, navigation capabilities remained robust. Results are detailed in Table XIII. These optimizations make NaVILA deployable directly on the robot, which will significantly eliminate image transmission time. We leave this as future work.

Fig. 12: Obstacle avoidance screenshots. Locomotion policy can ensure collision-free in the face of high grass, certain transparent glass, and large objects under strong sunlight. The policy presents robustness on sand and grass terrains.

TABLE XIII: NaVILA quantization results. The computational cost is tested on RTX 4090 with 1737 context tokens and 10 generated tokens, using a sample from R2R-CE as the test case.

	Computational Cost		R2R Val-Unseen			
	Total Latency (ms) ↓	GPU Memory (GB) ↓	NE ↓	OS ↑	SR ↑	SPL ↑
NaVILA (FP16)	594.58	18.5	5.37	57.6	49.7	45.5
NaVILA (W4A16)	**367.80**	**8.6**	5.66	56.8	48.2	43.6

E. Limitations

In Figure 13, we highlight a failure case in the real world where the robot initially follows the prompt but ultimately fails to reach the bedroom. This failure arises from the robot's inability to perform effective error correction when deviations occur. To further enhance performance, improving generalizability and spatial understanding is key. One potential direction is larger-scale training on more realistic simulations, which could provide more diverse navigation scenarios and error recovery cases. Additionally, incorporating explicit reasoning data during training could help the model better anticipate and correct mistakes.

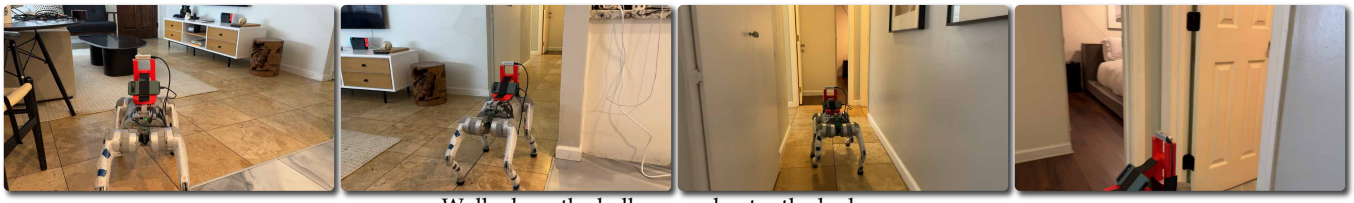


Fig. 13: Failure case of NaVILA.