AdaVLN: Towards Visual Language Navigation in Continuous Indoor Environments with Moving Humans

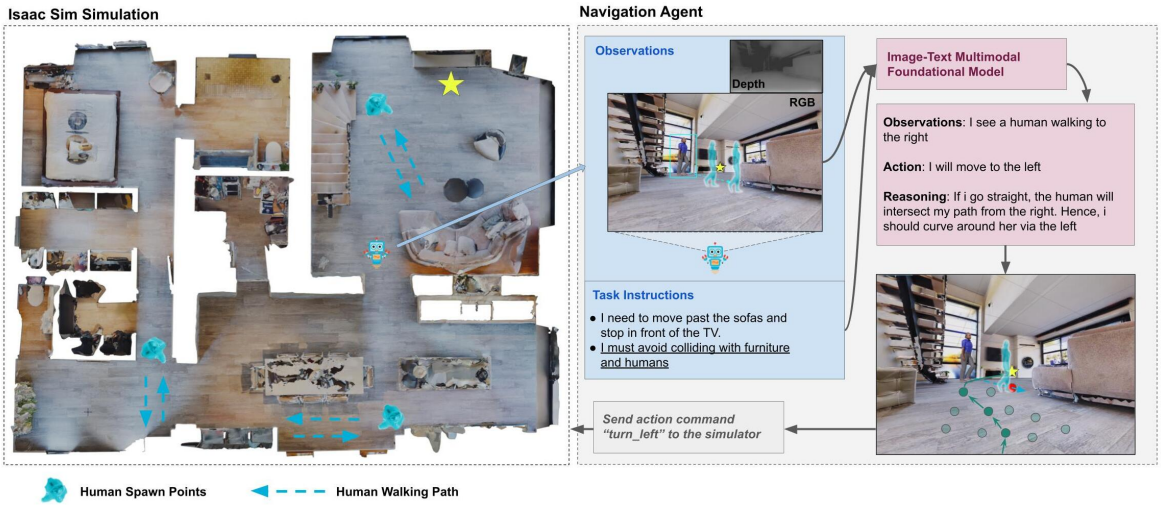

Figure 1: AdaVLN: Given natural language instructions, the robot is tasked to navigate in a continuous indoor space while avoiding collisions with moving humans. At each navigation step, the navigation agent receives an egocentric forward-facing RGB-D observation of its environment. Our baseline agent then sends the observations and task instructions to a large language model, which reasons a set of actions and the actions are then sent to the robot to execute the next navigation step.

Abstract

Visual Language Navigation (VLN) is a task that challenges robots to navigate in realistic environments based on natural language instructions. While previous research has largely focused on static settings, real-world navigation must often contend with dynamic human obstacles. Hence, we propose an extension to the task, termed Adaptive Visual Language Navigation (AdaVLN), which seeks to narrow this gap. AdaVLN requires robots to navigate complex 3D indoor environments populated with dynamically moving human obstacles, increasing task complexity and realism. To support exploration of this task, we also present AdaVLN simulator and AdaR2R datasets. The AdaVLN simulator enables easy inclusion of fully animated human models directly into common datasets like Matterport3D. We also introduce a "freeze-time" mechanism for both the navigation task and simulator, which pauses world state updates during agent inference, enabling fair comparisons and experimental reproducibility across different hardware. We benchmark several baseline models in simulation and real environments, analyze the unique challenges of AdaVLN, and show its potential to narrow the sim-to-real gap in VLN research.

Index Terms: Visual language navigation, embodied AI, agent, dynamic obstacles

1 Introduction

Visual navigation in indoor environments, a key topic in Embodied AI, studies an agent's ability to follow natural language instructions to reach a goal in unknown spaces. Solving this task requires the agent to (1) perceive and remember its surroundings, (2) interpret language instructions, and (3) plan actions that integrate spatial memory and language understanding [45].

While the premise of this task is straightforward, different variants have been introduced over the years, which can be broadly classified based on *communication complexity* (single/multi-turn interaction), *task objective* (action/goal-directed), and *action space* (discrete/continuous spaces) [45, 11, 5, 32]. Within this framework, Visual Language Navigation (VLN) is typically single-turn and action-directed, with discrete or continuous action spaces depending on the variant [5, 17]. VLN in continuous indoor environments (VLN-CE) has gained attention for real-world uses such as home robotics [17], yet its datasets and simulators remain largely static, lacking moving obstacles and evolving layouts. In reality, humans and other agents move within the same space, requiring robots to predict dynamic obstacle motion and adjust routes—capabilities critical in related tasks like SOON, HANNA, VLNA, and VDN [46, 43, 22, 29, 36].

To narrow this gap, we introduce Adaptive Visual Language Navigation (AdaVLN), an extension of VLN-CE that incorporates moving human obstacles into Habitat-Matterport3D [31]. We further introduce a freeze-time mechanism that pauses simulation during agent inference, enabling fair evaluation across hardware with different processing speeds. Specifically, we introduce the following two tools to enable research in this topic:

- AdaSimulator: A simulator offering physics-based 3D environments with dynamically moving obstacles like humans and accurate mobile robot movements. AdaSimulator is based on IsaacSim [20] and is built to be compatible with Matterport3D environments [7] and supports easy customisation of human spawn points, and pathing logic.
- AdaR2R: A sample variant of the R2R [5] and Matterport3D datasets that includes spawn points and trajectories for dynamic obstacles, adding another layer of realism to the navigation task.

Finally, we conduct experiments with several baseline models to evaluate our new task, analyzing the impact of these additional complexities on agent behavior and performance.

Figure 2: A robot navigates in a simulated dynamic environment with moving human obstacles.

2 Related Works

2.1 Visual Language Navigation

Over the years, the VLN task has advanced to better bridge simulation and real-world deployment. The original VLN task and Room-to-Room (R2R) dataset [5] required an agent to follow a single natural-language instruction to reach a goal in static 3D environments, using 360° RGB-D panoramas to select discrete neighbouring nodes, and led to the creation the Matterport3D dataset and simulator [7]. This simulator later supported related static-environment tasks such as SOON [46] and REVERIE [29]. Subsequent dataset extensions, including R4R [15] and RxR [18], increased instruction diversity and difficulty, while tasks such as Embodied Question Answering [43] and Vision-and-Language Navigation with Actions (VLNA) [22] extended objectives beyond navigation to question answering and action execution.

Building on these foundations, Habitat Sim [28, 35, 34] and the Habitat-Matterport3D mesh datasets [31] enabled experiments in physics-enabled 3D environments. Krantz et al. [17] extended VLN to continuous action spaces (VLN-CE), where robots execute low-level movements , spurring RxR-Habitat competitions at CVPR [10, 3] and follow-up research on world-state modelling [2, 39, 42] and long-term planning [41, 38]. Hong et al. [13] further revealed the complementary nature of discrete and continuous variants. More recently, Li et al. [19] proposed the Human-Aware MP3D (HA3D) simulator and Human-Aware R2R dataset, introducing perspective-specific animated humans and collision statistics into evaluation, thereby increasing realism and safety considerations for VLN.

2.2 Collision Avoidance during Navigation

Collision avoidance is a well-researched topic in robotics, where local or offline path-planning is crucial for safe operation in unknown environments. Effective planning requires anticipating how the surrounding world will evolve along candidate trajectories to avoid obstacles in advance. Classical approaches employ Velocity Obstacles [9] and their multi-agent extension, Reciprocal Velocity Obstacles, while motion cues from RGB-D sensing have also been explored [12]. More recent work combines reinforcement learning with representations of dynamic obstacles as variable-size ellipsoids and incorporates agent–human interaction dynamics [37, 27, 6], often supported by grid-, graph-, or 3D-based environment models for richer spatial reasoning [8, 40, 14].

Concepts from these studies have been adopted in VLN to enhance planning and obstacle prediction. DREAMWALKER [38] employs mental simulations to forecast environments along prospective paths; [2] introduces dynamic topological planning for obstacle avoidance; and Jeong et al. [16] develop the VLN-CM agent, which predicts occupancy maps from depth data to guide navigation.

3 Adaptive Visual Language Navigation

The existing VLN and VLN-CE tasks are largely focused on navigation in static environments, and do not explicitly define scenarios where dynamic obstacles like moving humans are present. To provide realistic, human-populated environments, we introduce Adaptive Visual Language Navigation (AdaVLN) (Figure 1), an extension to the VLN-CE task. In AdaVLN, we first propose the task objectives and define the actions of robots. We further introduce a freeze-time mechanism that pauses simulation during agent inference, enabling fair evaluation across hardware with different processing speeds. To support exploration of this task, we also present AdaVLN simulator and AdaR2R datasets. The AdaVLN simulator offers physics-based 3D environments with dynamically moving obstacles like humans and accurate mobile robot movements. The AdaR2R datasets includes spawn points and trajectories for dynamic obstacles, adding another layer of realism to the navigation task.

3.1 Task Description

Building upon the VLN-CE task, AdaVLN sets robots in Matterport3D environments with continuous action spaces. Each episode starts at time $t_0 = 0$ with the robot initialized at position (X_0, θ_0) and tasked to navigate to a goal position X_G by following natural language instructions provided at the start. A key addition in AdaVLN is the inclusion of dynamic obstacles — in the form of humans — and an emphasis on collision avoidance. The states of these obstacles, denoted (X'_t, θ'_t), are continuously updated as they move along NavMesh paths between pre-defined waypoints in the AdaR2R dataset. Robots are required to avoid collision with both static obstacles and the dynamic obstacles (Figure 2).

3.2 Observations/Actions of Robots

At each navigation step t, the robot observes an egocentric front-facing view of its surroundings in the form of an RGB-D image [19], as seen in Figure 3. Based on this observation and its current state, the robot can selects one of four actions:

1. Turn left by 15 degrees at 30 degrees/s
2. Turn right by 15 degrees at 30 degrees/s
3. Move forward 0.25 meters at 0.5m/s
4. Stop

As action duration can influence navigation performance, the robot was configured to move at 0.5 m/s and rotate at 30°/s, ensuring a uniform execution time of approximately 2s per action.

The 'stop' command indicates the end of an episode, upon which the robot and simulation stops. The agent's performance is then evaluated based on its final state (X_f, θ_f) and the path it took, represented as (X_t, θ_t) for $t \in [0, T_f]$, where T_f is the final time step. Each episode is capped at 50 steps, after which the simulation terminates if the goal is not reached.

3.3 Freeze-Time

As dynamic obstacles are updated at every simulation tick, variations in hardware capability and inference latency may bias experimental outcomes. To achieve hardware-agnostic evaluation, an optional "Freeze-Time" mode will be offered that suspends world-state updates while the agent computes its next action; this option can be disabled when the inference speed is intended to be part of the evaluation.

Figure 3: Top: RGB observations, Bottom: Depth observations provided to agent. Note that the depth observations have been restricted to a range between 0 and 10 in this image for clarity.

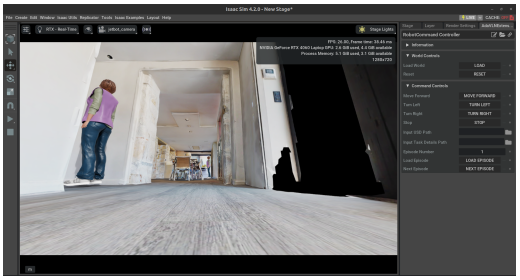

Figure 4: AdaSimulator's GUI Extension in Isaac Sim

3.4 AdaSimulator

AdaSimulator is implemented as a standalone extension to IsaacSim, leveraging its physics engine and RTX Renderer [25]. The simulator automatically sets up all necessary environment components when loading a scene:

- Sets up collider meshes for the static obstacles
- Spawns a Jetbot at (X_0, θ_0)
- Sets up camera render products for generating observations
- Loads environment lighting rigs
- Loads humans at (X_0', θ_0') and sets up their animation graphs

All simulation scenarios employ a two-wheeled NVIDIA Jetbot [24] driven by differential controllers for physics-based motion. Egocentric observations are generated with IsaacSim's Replicator Core from the Jetbot-mounted camera. Dynamic human obstacles are incorporated using a customized `omni.anim.people` extension [23]. An ROS 2 interface enables agents to access RGB-D observations and issue control commands.

The simulator can be run in GUI mode for full visibility of the navigation episodes and manual input of robot commands, or in headless mode for optimal training speed (Figure 4).

3.5 AdaR2R (Sample)

AdaR2R (Sample) provides 9 navigation episodes across 3 HM3Dv2 scenes [30, 21], with representative snapshots shown in Figure 5 [31]. It extends the original R2R format with specifications for human spawn points, path waypoints, and motion parameters. Each episode contains 1–2 moving humans whose waypoints deliberately intersect critical straight-line connections in the reference path. These obstacles are transient: alternative detours exist or the humans eventually vacate the obstructed areas.

The tasks are purposely made to be simple as the focus is on the human obstacles, with an average geodesic distance for each navigation episode is 5.84 meters. As a sample, it serves as a reference for future works to establish new task variants of existing room-to-room datasets. The environment and robot both use triangular collider meshes with default offset values determined by IsaacSim.

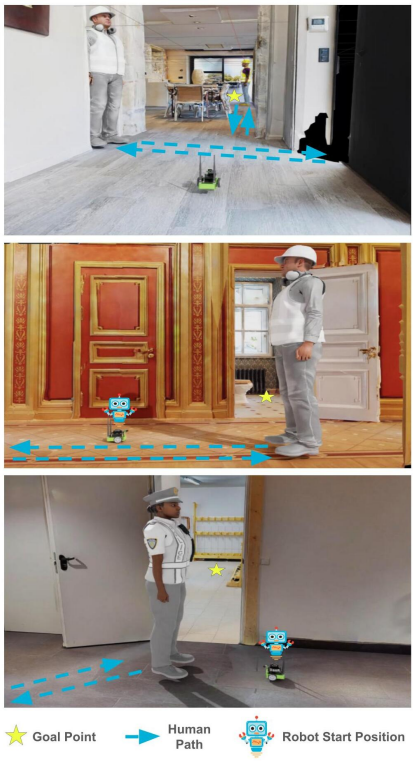

Figure 5: The 9 navigation episodes were conducted in 3 HM3Dv2 scenes. Humans loop along the indicated paths infinitely throughtout a navigation episode. Note that the paths have been deliberately chosen to interfere with the optimal path the robot would take.

4 Experiments

4.1 Evaluation Protocol

To demonstrate the task and simulator, we evaluate a baseline agent's ability to reach goals while avoiding humans and static obstacles. While established VLN metrics primarily assess navigation performance [4, 5, 44], our emphasis on a new simulation framework leads us to focus on collision statistics. We report Navigation Collisions (NC), the fraction of time the agent remains in contact with any obstacle, further decomposed into Human Navigation Collisions (HNC) and environmental collisions, and complement these with qualitative analyses of agent observations and actions across representative episodes. Given the shorter geodesic distances than R2R and R2R-CE, each episode is limited to 50 navigation steps. The NVIDIA Jetbots are configured with differential controllers using a wheel radius of 0.035 m and a wheel base of 0.1 m.

4.2 Baseline GPT Agent

We evaluate a simple agent built on the GPT-4o-mini multi-modal model [26]. At each navigation step, the robot summarizes semantic observations from RGB inputs (down-scaled from 1280×720 to 640×360), formulates a long-term plan, reviews prior steps, and selects the next action. Predictions are limited to 200 output tokens per step. Implementation details are provided in the project repository.

Tests are conducted zero-shot with no prior training of any sort on our dataset. As seen in Figure 6 and Table 1, collision rates in general are high, due to poor environmental parsing capabilities of

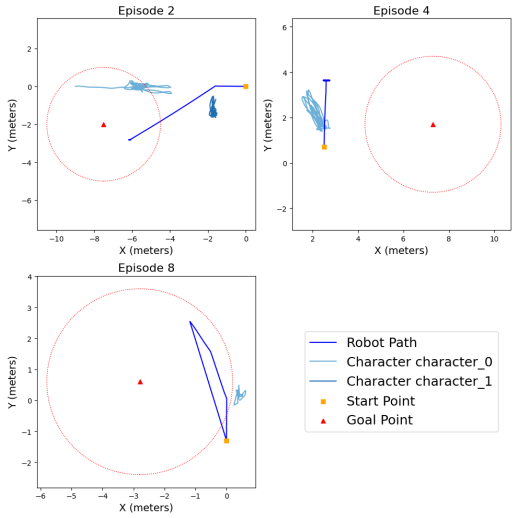

Figure 6: Top: Sample of paths (represented by lines) taken by robots and humans during simulation. Coordinate origins are based on X-Y provided in MP3D GLB files which have been scaled to 1 unit : 1 meter. In cases where the robot's line moves back-and-forth around a point, the robot has gotten stuck in collision with a wall.

our agent. In particular, we note that our agent frequently makes hallucinating observations which include:

- Stating that paths ahead are clear even if they are facing a wall
- Stating that there are no humans or obstacles in front of them even if there are
- Hallucinating the instruction's objects in front of them

Because AdaSimulator models full robot and environment physics, recovering from wall collisions is considerably harder than in HabitatSim. Instead of sliding along walls, a robot may tip backward when pushing against a wall or become unable to turn, as no reverse action is defined. Once immobilized, escape is effectively impossible, adding a realistic challenge absent in HabitatSim, which permits wall sliding.

Although human collisions constitute a small proportion of the total collisions, this is primarily because humans continue on their paths and exit the collision zone after contact. As shown in 6, the agent makes little effort to navigate around human obstacles. We hypothesize that this behavior is due to the lack of realism in the human 3D models, causing the foundational model to fail to recognize them as obstacles.

4.3 Real-World Experiment Preparation

We also conducted experiments in real environments with a TurtleBot3 Waffle robot [1] and a desktop workstation (host) running the agent. Host–robot communication used ROS 2 (Foxy Fitzroy) with Fast DDS over Wi-Fi. The robot base supports maximum linear and angular speeds of 0.26 m/s and 1.82 rad/s. High-level computation ran on a Raspberry Pi 4 (2 GB RAM), while low-level motor control and sensor I/O were handled by OpenCR controller [33]. Perception relied on a Stereolabs ZED 2 stereo camera using the official ROS zed-ros2-wrapper, streaming synchronized 1280×720 RGB images per eye at 60 Hz. Our analyses only used raw RGB frames. To match wireless bandwidth and latency, every 30th stereo pair was published to the host.

4.4 Procedure & Sim-to-Real Analysis

We evaluated the system in a laboratory mock-up comprising two static box-like obstacles, a target object (red balloon), and a randomly appearing pedestrian to induce dynamic occlusions. During each trial, the onboard client captured 115° front-facing RGB images at 60 Hz and transmitted the latest frame with a structured prompt (task instruction, safety rules, and state/action summary) to the AI agent host at every 0.5s (2Hz). The host returned a policy specifying one of five discrete actions and accompanying reasoning. Upon receiving an action, the robot executed the corresponding motion primitive (e.g., move_forward→ $v = 0.1$ m/s; turn_left/right→ $\omega = \pm 0.5$ rad/s; stop→ $v = \omega = 0$) and re-evaluated at 2 Hz, ensuring compliance with task objectives and safety constraints. We conducted 20 trials with pedestrian randomly moving in front of the robots and there was no collision detected.

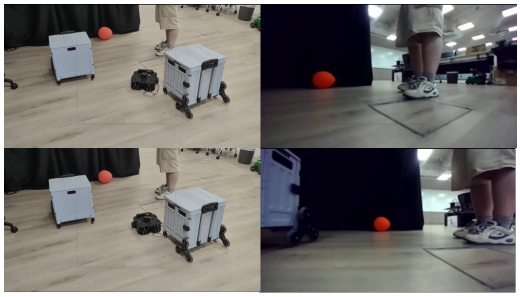

Figure 7: Third-person (left column) and onboard camera (right column) views of the laboratory trial with barriers, target and moving people. Top: robot and scene just before the AI agent issues the command. Bottom: robot and view after the AI agent issues the command.

Episode	Environmental NC	Human NC	Combined NC
1	0.77	0.01	0.78
2	0.69	0.01	0.70
3	0.91	0.01	0.91
4	0.93	0.00	0.93
5	0.86	0.00	0.86
6	0.76	0.00	0.76
7	0.00	0.00	0.00
8	0.71	0.08	0.78
9	0.00	0.00	0.00
Average	**0.63**	**0.01**	**0.64**

Table 1: Normalized Collision (NC) values across different episodes. The combined navigation collision measures the total amount of timesteps a robot spends in a navigation episode while in collision with any object in the scene. Environmental and Human NC only considers collisions with static obstacles (like furnitures/wall) or humans respectively.

Figure 7 shows a representative laboratory trial where the robot is tasked to reach a red balloon. The left column shows third-person views and the right column shows onboard camera images. The top row shows photos captured before the AI agent issues a `turn_left` command and the bottom row shows photos captured after the robot executes the received instructions. In this episode, the robot successfully avoids a moving pedestrian and a static barrier while advancing toward the target (a red balloon).

5 Conclusion and Future Work

We presented AdaVLN, which extends the VLN-CE problem towards agent/robot navigation in dynamic environments featuring moving humans as dynamic obstacles. Alongside this, we introduced AdaSimulator, an extension of IsaacSim that facilitates the setup of fully physics-enabled simulations with realistic robots and animated 3D humans. Our baseline experiments demonstrate that the added complexity of our simulator enables more realistic evaluations and highlights the potential challenges of the new task. We aim to expand on this work by refining the simulation environment, generalizing the task formalization to broader dynamic environments, and developing agents capable of effectively navigating these complex scenarios.

References

[1] R. Amsters and P. Slaets. Turtlebot3 as a robotics education platform. In *Robotics in Education*, pp. 170–181. Springer International Publishing, 2019. doi: 10.1007/978-3-030-26945-6
$_{1}$64

[2] D. An, H. Wang, W. Wang, Z. Wang, Y. Huang, K. He, and L. Wang. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. 2024. 2

[3] D. An, Z. Wang, Y. Li, Y. Wang, Y. Hong, Y. Huang, L. Wang, and J. Shao. 1st place solutions for rxr-habitat vision-and-language navigation competition (cvpr 2022). 2022. 2

[4] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, and A. R. Zamir. On evaluation of embodied navigation agents. (arXiv:1807.06757), July 2018. arXiv:1807.06757 [cs]. 3

[5] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. v. d. Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. (arXiv:1711.07280), Apr. 2018. arXiv:1711.07280 [cs]. 1, 2, 3

[6] M. Castillo-Lopez, S. A. Sajadi-Alamdari, J. L. Sanchez-Lopez, M. A. Olivares-Mendez, and H. Voos. Model predictive control for aerial collision avoidance in dynamic environments. In *2018 26th Mediterranean Conference on Control and Automation (MED)*, p. 1–6. IEEE, Zadar, Croatia, June 2018. doi: 10.1109/MED.2018.8442967 2

[7] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Nießner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3d: learning from rgb-d data in indoor environments. (arXiv:1709.06158), Sept. 2017. arXiv:1709.06158 [cs]. 1, 2

[8] A. Elfes. Occupancy grids: a stochastic spatial representation for active robot perception. (arXiv:1304.1098), Mar. 2013. arXiv:1304.1098 [cs]. 2

[9] P. Fiorini and Z. Shiller. Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7):760–772, July 1998. doi: 10.1177/027836499801700706 2

[10] Google. Rxr habitat, 2024. Accessed: 2024-11-17, https://ai.google.com/research/rxr/habitat. 2

[11] J. Gu, E. Stefani, Q. Wu, J. Thomason, and X. E. Wang. Vision-and-language navigation: a survey of tasks, methods, and future directions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, p. 7606–7623, 2022. arXiv:2203.12667 [cs]. doi: 10.18653/v1/2022.acl-long.524 1

[12] E. Herbst, X. Ren, and D. Fox. Rgb-d flow: Dense 3-d motion estimation using color and depth. In *2013 IEEE International Conference on Robotics and Automation*, pp. 2276–2282, 2013. doi: 10.1109/ICRA.2013.6630885 2

[13] Y. Hong, Z. Wang, Q. Wu, and S. Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. 2022. 2

[14] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. Octomap: an efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206, Apr. 2013. doi: 10.1007/s10514-012-9321-0 2

[15] V. Jain, G. Magalhaes, A. Ku, A. Vaswani, E. Ie, and J. Baldridge. Stay on the path: instruction fidelity in vision-and-language navigation. (arXiv:1905.12255), June 2019. arXiv:1905.12255 [cs]. 2

[16] S. Jeong, G.-C. Kang, J. Kim, and B.-T. Zhang. Zero-shot vision-and-language navigation with collision mitigation in continuous environment. (arXiv:2410.17267), Oct. 2024. arXiv:2410.17267 [cs]. 2

[17] J. Krantz, E. Wijmans, A. Majumdar, D. Batra, and S. Lee. Beyond the nav-graph: vision-and-language navigation in continuous environments. (arXiv:2004.02857), May 2020. arXiv:2004.02857 [cs]. 1, 2

[18] A. Ku, P. Anderson, R. Patel, E. Ie, and J. Baldridge. Room-across-room: multilingual vision-and-language navigation with dense spatiotemporal grounding. (arXiv:2010.07954), Oct. 2020. arXiv:2010.07954 [cs]. 2

[19] H. Li, M. Li, Z.-Q. Cheng, Y. Dong, Y. Zhou, J.-Y. He, Q. Dai, T. Mitamura, and A. G. Hauptmann. Human-aware vision-and-language navigation: bridging simulation to reality with dynamic human interactions. (arXiv:2406.19236), Nov. 2024. arXiv:2406.19236 [cs]. 2

[20] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac gym: High performance gpu-based physics simulation for robot learning. 2021. 1

[21] Meta AI. Habitat-matterport 3d dataset v2 (hm3dv2), 2023. Accessed: 2025-09-15, https://aihabitat.org/datasets/hm3d-v2/. 3

[22] K. Nguyen, D. Dey, C. Brockett, and B. Dolan. Vision-based navigation with language-based assistance via imitation learning with indirect intervention. (arXiv:1812.04155), Apr. 2019. arXiv:1812.04155 [cs, stat]. 1, 2

[23] NVIDIA. Omni.anim.people, 2024. Accessed: 2024-11-17, https://docs.omniverse.nvidia.com/isaacsim/latest/features/warehouse_logistics/ext_omni_anim_people.html. 3

[24] NVIDIA Corporation. Jetbot: An open-source robot based on nvidia jetson nano. https://github.com/NVIDIA-AI-IOT/jetbot. Accessed Sep. 14, 2025. 3

[25] NVIDIA Corporation. Omniverse rtx renderer, 2025. Accessed: 2025-09-15, https://docs.omniverse.nvidia.com/materials-and-rendering/latest/rtx-renderer.html. 3

[26] OpenAI. Gpt-4o mini: advancing cost-efficient intelligence, 2024. Accessed: 2024-11-17, https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/. 3

[27] M. Pfeiffer, S. Shukla, M. Turchetta, C. Cadena, A. Krause, R. Siegwart, and J. Nieto. Reinforced imitation: Sample efficient deep reinforcement learning for map-less navigation by leveraging prior demonstrations. (arXiv:1805.07095), Aug. 2018. arXiv:1805.07095 [cs]. 2

[28] X. Puig, E. Undersander, A. Szot, M. D. Cote, R. Partsey, J. Yang, R. Desai, A. W. Clegg, M. Hlavac, T. Min, T. Gervet, V. Vondruš, V.-P. Berges, J. Turner, O. Maksymets, Z. Kira, M. Kalakrishnan, J. Malik, D. S. Chaplot, U. Jain, D. Batra, A. Rai, and R. Mottaghi. Habitat 3.0: A co-habitat for humans, avatars and robots. 2023. 2

[29] Y. Qi, Q. Wu, P. Anderson, X. Wang, W. Y. Wang, C. Shen, and A. v. d. Hengel. Reverie: remote embodied visual referring expression in real indoor environments. (arXiv:1904.10151), Jan. 2020. arXiv:1904.10151 [cs]. 1, 2

[30] S. K. Ramakrishnan, A. Gokaslan, A. Clegg, E. Undersander, A. Galindo, A. X. Chang, M. Savva, and D. Batra. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12876–12884, 2021. 3

[31] S. K. Ramakrishnan, A. Gokaslan, E. Wijmans, O. Maksymets, A. Clegg, J. Turner, E. Undersander, W. Galuba, A. Westbury, A. X. Chang, M. Savva, Y. Zhao, and D. Batra. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. (arXiv:2109.08238), Sept. 2021. arXiv:2109.08238 [cs]. 1, 2, 3

[32] S. Raychaudhuri, D. Ta, K. Ashton, A. X. Chang, J. Wang, and B. Bucher. Nl-slam for oc-vln: Natural language grounded slam for object-centric vln. (arXiv:2411.07848), Nov. 2024. arXiv:2411.07848 [cs]. 1

[33] ROBOTIS. *OpenCR1.0 (Open-source Control Module for ROS) e-Manual*, 2025. Accessed: 2025-09-15. 4

[34] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 2

[35] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2

[36] J. Thomason, M. Murray, M. Cakmak, and L. Zettlemoyer. Vision-and-dialog navigation. (arXiv:1907.04957), Oct. 2019. arXiv:1907.04957 [cs]. doi: 10.48550/arXiv.1907.04957 1

[37] X.-T. Truong and T. D. Ngo. Toward socially aware robot naviga-

tion in dynamic and crowded environments: A proactive social motion model. *IEEE Transactions on Automation Science and Engineering*, 14(4):1743–1760, Oct. 2017. doi: 10.1109/TASE.2017.2731371 2

[38] H. Wang, W. Liang, L. Van Gool, and W. Wang. Dreamwalker: Mental planning for continuous vision-language navigation. (arXiv:2308.07498), Aug. 2023. arXiv:2308.07498 [cs]. 2

[39] T. Wang, Z. Wu, F. Yao, and D. Wang. Graph based environment representation for vision-and-language navigation in continuous environments. 2023. 2

[40] T. Wang, Z. Wu, F. Yao, and D. Wang. Graph based environment representation for vision-and-language navigation in continuous environments. (arXiv:2301.04352), Jan. 2023. arXiv:2301.04352 [cs]. 2

[41] Z. Wang, X. Li, J. Yang, Y. Liu, J. Hu, M. Jiang, and S. Jiang. Lookahead exploration with neural radiance representation for continuous vision-language navigation. 2024. 2

[42] Z. Wang, X. Li, J. Yang, Y. Liu, and S. Jiang. Gridmm: Grid memory map for vision-and-language navigation. 2023. 2

[43] E. Wijmans, S. Datta, O. Maksymets, A. Das, G. Gkioxari, S. Lee, I. Essa, D. Parikh, and D. Batra. Embodied question answering in photorealistic environments with point cloud perception. (arXiv:1904.03461), Apr. 2019. arXiv:1904.03461 [cs]. 1, 2

[44] L. Yue, D. Zhou, L. Xie, F. Zhang, Y. Yan, and E. Yin. Safe-vln: Collision avoidance for vision-and-language navigation of autonomous robots operating in continuous environments. (arXiv:2311.02817), Apr. 2024. arXiv:2311.02817 [cs]. 3

[45] Y. Zhang, Z. Ma, J. Li, Y. Qiao, Z. Wang, J. Chai, Q. Wu, M. Bansal, and P. Kordjamshidi. Vision-and-language navigation today and tomorrow: a survey in the era of foundation models. (arXiv:2407.07035), July 2024. arXiv:2407.07035 [cs]. 1

[46] F. Zhu, X. Liang, Y. Zhu, X. Chang, and X. Liang. Soon: scenario oriented object navigation with graph-based exploration. (arXiv:2103.17138), Oct. 2021. arXiv:2103.17138 [cs]. 1, 2