HARDNAV - Simulator for Benchmarking Robust Navigation and Place Recognition in Large, Confusing and Highly Dynamic Environments

Tomáš Musil, Matěj Petrlík, Martin Saska

***Abstract*— We present a novel simulator called HARDNAV for developing and benchmarking robust vision-based autonomous navigation and place recognition. The simulator is designed to effortlessly simulate challenging scenarios and environmental features for single-mission autonomy, such as dynamic objects and lights, featureless areas or sensor corruption, and for long-term autonomy, such as visibility and topology changes and large-scale unstructured environments. Additionally, we propose replicable benchmarks of active place recognition, and of multi-session navigation, specifically for a kidnapped robot return home mission type, and discuss other challenging benchmarks possible in our simulator. We opensource the code for the simulator and provide scripts and tutorials to easily design multi-session experiments. We hope the simulator will help the robotics and AI community to develop truly robust spatial intelligence methods.**

Code— https://github.com/MrTomzor/navigation_unity_testbed

I. INTRODUCTION

Autonomous navigation, place recognition and SLAM are aspects of an emerging more general field - spatial/embodied artificial intelligence. These aspects are essential for the development of any kind of embodied agents that can perform complex and meaningful tasks in the physical world. They have been heavily researched and reviewed in the last several decades [1]–[3], but the existing methods are often still not robust enough for the real world.

For example most state-of-the-art methods, especially for SLAM, still make very strong and limiting assumptions about the world — that it is mostly static, and that global metric localization is achievable at all times. Some SLAM algorithms try to relax the metric localization requirements, such as [4], but those rely on purely image-based appearance matching for place recognition, which can fail for example under severe illumination/visibility changes.

Navigation also suffers from these assumptions, and even learning-based methods often incorporate a perfect position sensor into its inputs [5]. There are some methods that attempt to break free of this assumption [6], but these are often tailored to specific edge cases and are not a general solution.

In contrast, living organisms are able to navigate changing, ambiguous, large-scale environments with ease. Despite the discovery of grid cells [7], which seem to form an internal metric system, humans and animals can navigate environments where metric localization is unattainable (e.g., navigating abstract conceptual spaces such as a social hierarchy [8]). They can also quickly change their cognitive maps in case of drastic structural changes in the environment to adapt without needing hundreds of hours of retraining time [8] [9]. There is, therefore, still many open questions and challenges in spatial intelligence to answer before spatial artificial intelligence methods are anywhere near the level of robustness and efficiency of living organisms.

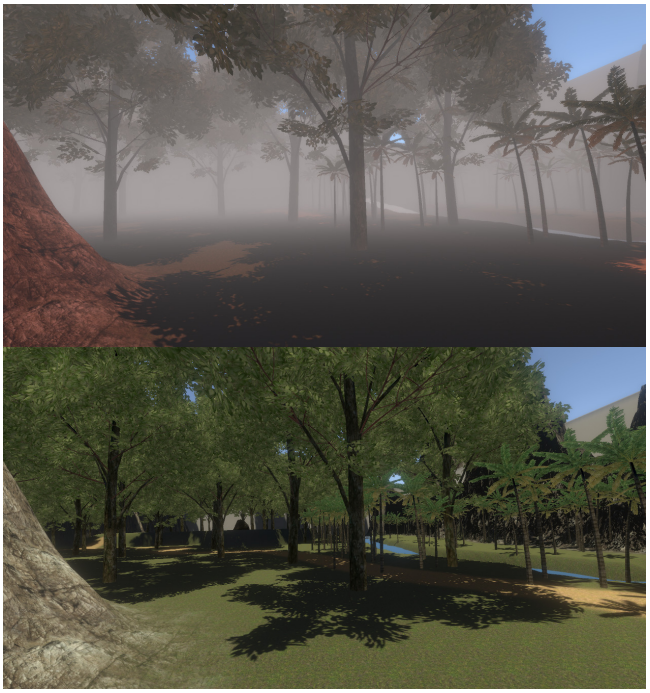

Fig. 1: A viewpoint from the *Forest1* environment featured in HARDNAV seen under heavy appearance change between sessions — color and lighting angle variation and fog. All of these settings are toggleable per session by a service call without having to edit world files.

In this paper, we present a simulator specifically designed to simulate the difficult situations and environments in which current spatial intelligence methods fail, such as illumination and visibility changes, self-similar unstructured large-scale environments, featureless areas, drastic environment changes between sessions and high numbers of dynamic objects and lighting effects. We hope this will help researchers quickly test their methods against problems such as wrong loop closures, odometry/scale drift, memory capacity and tractability problems.

Additionally, we propose a specific replicable benchmark for the task of autonomous navigation that uses the

Authors are with the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, 166 36 Prague 6, {musilto8|matej.petrlik|martin.saska}@fel.cvut.cz

Digital Object Identifier (DOI): see top of this page.

HARDNAV simulator and discuss additional possible uses for the simulator, such as synthetic dataset generation for heavy session change.

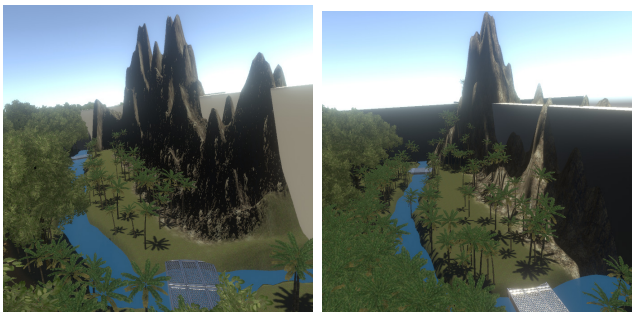

Fig. 2: An example of perceptual aliasing in the *Forest1* environment taken during constant lighting conditions - these two areas of the map are visually very similar, differing mainly in the outline of the mountains and in the distribution of palm/alder trees on the other bank, but lie in opposite corners of the map.

II. RELATED WORKS

A. Simulators

Currently, there is a vast amount of simulators for robotics applications that feature different motion and sensor models in different 3D environments, such as Flightmare [10] which is tailored mainly for development of control and planning methods for quadrotors and has a modular structure with a high-fidelity physics engine, CARLA [11] which features large-scale road traffic and weather change and [12] for quadrotor simulation with the high-fidelity NVIDIA PhysX engine.

In the machine learning community, there are also many simulators available for embodied AI aresearch - for example Habitat [13] and RoboTHOR [14]. These mostly focus on the topic of "vision-language navigation" in which deep learning models are trained to perform tasks specified by humans in natural language [15].

The majority of these simulators still feature only static, small-scale, structured scenes, with the major exception being CARLA [11], which is however designed for cars moving in the highly structured and navigable domain of roads. In contrast, HARDNAV does not offer high-fidelity dynamics nor super-realistic rendering because it is focused on offering large-scale, unstructured, confusing, dynamic environments and the ability to easily modify the configuration of the world and design multi-session tasks.

B. Benchmarks

Perhaps the most impactful recent benchmark of navigation capabilities was the Darpa SubT Challenge [16]. In the challenge, robot teams had to (semi-autonomously in the systems track, and fully autonomously in the virtual track) explore unstructured subterranean environments, detect and determine precise metric location of pre-defined objects, with each team having a limited amount of guesses and receiving one point per correct object report, meaning correct class label and position error less than 5 meters. The challenge pushed the limits of current methods, and featured difficult environmental conditions, such as featureless areas, non-reflective surfaces, dynamic obstacles, smoke or water. In the end, nearly all teams relied on multi-row LiDARs for precise localization. In our benchmarks, we try to relax the goal of precise metric localization, but keep the challenging environmental conditions.

Fig. 3: A showcase of a part of the *ScifiBase1* world, with 2 types of robots that wander around the area randomly using Unity's built-in navmesh functionalities, multiple metal containers that can be randomized at the start of each session, and light rain with fog and clouds

Another approach at benchmarking robust navigation is [17] where the authors created a framework in which they apply visual corruptions (e.g. lens cracks) and dynamic corruptions (e.g. forward motion leading to slight turning) to the agent in their environment and opensource their code to benchmark robustness of embodied navigation agents. Compared to our environment, their work and other works [18], [14] feature procedurally generated environments of household interiors, while our simulator focuses more on large-scale unstructured and dynamic environments.

There are also several datasets available for benchmarking robust perception, such as OIVIO [19] which features exploration of a mineshaft with large amounts of motion blur, or 4seasons [20] which captures one environment from a car at multiple times in a year. These datasets are undoubtedly useful for benchmarking robust localization and mapping. However, robust navigation, which is often the end goal on top of SLAM, cannot be fully evaluated using only datasets, and requires either expensive real-world experiments, or using a simulator, which we try to provide.

III. HARDNAV SIMULATOR

The HARDNAV simulator uses Unity both for rendering and for physics simulation, since we do not strive for high-fidelity control or dynamics simulation, but rather large-scale long-term navigation tasks. We have chosen Unity over the widely used Gazebo simulator primarily for its ease of modifying scenes, and its superior visual fidelity, as discussed in [21].

Challenge	Examples	Expected negatively affected methods
Visual corruption	Dust particles, *leaves/dirt lifted from ground*, rain, *snow*, motion blur, *lens dirt*, big exposure changes	all of vision
Dynamic objects	Ground and air robots randomly moving around environment, grass and trees swaying in the wind, clouds, moving water	odometry drifting if many objects nearby, faulty object-based place recognition
Dynamic lights	Flickering lights, *fires*, many light sources on dynamic objects, light source on controlled robot	purely visual odometry drifting due to many outliers or perceived motion
Illumination and visibility changes	Global directional light of varying intensity, color, angle; fog of varying color and density	faulty visual place recognition, odometry having low amount of features
Small structural change across sessions	Medium-size objects' positions being randomized, *trees falling down between sessions*	wrong place recognition, relocalization, teach-and-repeat navigation
Drastic structural change across sessions	Passages being blocked, *buildings being built*, distant landmarks disappearing, *snowfall with varying height*	wrong place recognition, relocalization, teach-and-repeat navigation
Robot affecting scene	*Leaving footprints/tracks*, moving objects on collision	place recognition, relocalization
Featureless areas, transparent objects	textureless corridors (no visual features), straight smooth corridor/wide open area (no depth features), fences, glass walls	odometry drift, depth estimation failures in case of vision
Perceptual aliasing / self-similarity	Many areas having a very similar appearance appearance-wise, geometry-wise or both. Medium in *Forest1*, heavy in *ScifiBase1*	relocalization, loop closure
Scale variation	Having both small areas (buildings, small corridors) and comparably larger areas (vast outdoors, big rooms) in one environment	fixed-resolution mapping/navigation (e.g. voxel-based maps)
Large environment scale	The environments are approx. 2km wide with robots being approx. 1m wide	SLAM memory problems, difficulty learning with deep learning methods

TABLE I: The challenges for place recognition and navigation implemented (and planned, in italics) in HARDNAV. The upper portion of the challenges in the table can all be individually enabled or disabled in the session config YAML string for each session. The lower half challenges are caused by the design of the individual environments and are not toggleable.

The user-friendly interface of the Unity editor allows easy creation of new environmental behaviors (e.g. adding moving agents, automatic doors, changing textures etc.) through simple C# scripts, and also creating or modifying environments. It is also important to mention that Unity has an asset store with a plethora of free assets for world building, visual effects and potential agents. Overall, we believe that the potential of this game engine for creating challenging and useful robotics scenarios has not yet been fully explored and that HARDNAV might motivate this exploration.

Currently, the simulator features two worlds: one very open and visually rich - *Forest1* shown in Fig. 1 and one with a more labyrinth-like structure and barren - *ScifiBase1* shown in Fig. 4, both of which support changing of visibility conditions and toggling of the challenges specified in Table I through a ROS service by passing a YAML config string.

A. Implemented Agents and Sensors

As of now, the simulator offers 2 types of agents - an idealized space/underwater/flying robot with full force+torque or linear+angular velocity control, and a simple car robot. Traversability and control for both agents should be easy to implement, as it is not the focus of the simulator.

The sensors implemented so far are an RGB camera (can have any number of these on a robot) and an IMU sensor. Sensors of other modalities, such as a depth camera, an ultrasonic beam sensor, and bumpers, are planned. We have decided to not implement LiDARs as of yet, since we believe these make the navigation tasks far too easy or nudge people to take global metrically precise localization achievability as an assumption, which we propose to move away from. We only output the precise ground truth position of the robot to ROS as a means of visualizing navigation progress. However, it could be useful to add a very noisy GPS sensor for tasks like for example "search this approximate area of 100m around this GPS position in the forest".

B. Area Labels and World Configuration

Each environment in HARDNAV contains multiple *areas*, which are for now defined as oriented bounding boxes, containing multiple spawn points for the agent and having a text label. The simulator sends a string message to ROS at a fixed rate, containing the labels of all areas the robot is currently intersecting. We find this labeling realizable in the real world - for example when a robot is being "shown around" a new area, an operator could simply define an area by pressing a button and walking the robot along approximate boundaries of the desired area, and this specification does not require the robot to learn the human-language semantics of for example what "kitchen" means. The main point of this distinction is that we can then define missions in terms of *AreaNavigation*, as specified in [1] and not just metric positions in some given frame, and also to automatically evaluate missions such as "reach area A, avoid area B".

IV. ROBUST NAVIGATION AND PLACE RECOGNITION BENCHMARKS

In this section, we propose several ways of evaluating spatial navigation and spatial AI capabilities, and present the tools available in HARDNAV for conveniently running these evaluations.

A. Multi-Session Synthetic Dataset Generation

The most basic contribution of our simulator is that it can be easily used for generating synthetic datasets for visual place recognition under heavy inter-session change, simply by changing the configuration file of a given environment, and then manually driving the robot and recording a desired dataset. We hope this can be of benefit to the visual place

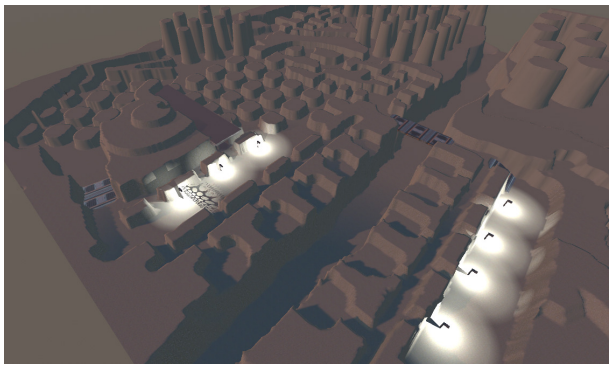

Fig. 4: Top view of the *ScifiBase1* environment featured in HARDNAV. Compared to *Forest1*, this environment is much more visually starved with no trees or distinct visual features, and with heavy self-similarity, forcing agents to learn the general, large-scale layout and being able to handle even multiple wrong place recognitions/loop closures.

recognition community and welcome any feedback on this matter.

B. Active Place Recognition under Heavy Environment Change

Since successful self-localization after being lost/kidnapped is a task necessary for succesful active navigation, we propose a task definition we call "active place recognition". Essentially, the task is that the robot should actively explore a previously seen part of the environment after being kidnapped, and in some given amount of time, output a label corresponding to which area (for example a bounding box in the world, as described in Sec. III-B) it is in. The focus here is on *active* place recognition, meaning that the robot should search for the environmental features that disambiguate the area from other previously seen areas. For example in the *Forest1* environment, there are two nearly identical environments which differ in subtle features, such as the shape of the nearby mountain and where the bridge leading from the island is placed, as shown in Fig. 2, and humans tend to immediately look for these features to disambiguate the situation in this task.

We believe this problem to be a vital subproblem for the task specified in the following section:

C. Multi-Session Return Home Challenge

For evaluating navigation capabilities, we use the taxonomy introduced by [1] and focus mostly on the AreaGoal navigation task - that is navigating to an area specified by a bounding box as defined in Sec. III-B. As the first main task, due to its importance in robotics, we chose to benchmark the navigation by spawning the robot at multiple previously seen areas and having it navigate to a "home" area.

In this task specification, the robot is spawned at previously seen zones, but not necessarily at previously visited waypoints. Previously seen in this context means seen in a pre-recorded dataset, which can either be collected through autonomous exploration or by manually driving the robot. We provide an example script for running this kind of navigation task, which can be modified to include not-seen areas and varying amounts of environment change between sessions to increase the difficulty.

D. Replicable benchmarks

To allow for replicable testing conditions for comparison of robust navigation approaches, we have constructed the environment so that without changing the YAML configuration provided through the ROS reset service, the environment will always be initialized to be the same. As an example, we are working on specifying a fixed benchmark in our environment, through a dataset and a script that initializes the individual trials of returning home from different areas visited in the dataset. We collect the dataset by driving the robot manually, and choose the maximum times for the return missions to be slightly higher than was the mission time of several human subjects.

V. FUTURE WORK AND DISCUSSION

The simulator is still a work in progress, and we are constantly adding new important features, shown on the github website. The simulator is flexible enough to allow creating any sort of multi-session navigation challenges, and we intend to formulate additional spatial tasks, such as searching for objects or covering a specified area. We will also likely connect the simulator with AI Gym [22], so that reinforcement learning approaches, ideally in combination with classical robotics approaches, can be used for tackling the simulated challenges. The priority of these potential features will be decided based on the feedback and suggestions from fellow researchers. We are also working on a quantitative evaluation of the performance of state-of-the-art SLAM and place recognition methods in the challenging environments available in HARDNAV.

We hope our simulator and benchmarks will help researchers develop robust spatial intelligence approaches that allow robots to work autonomously in the difficult, chaotic conditions of the ever-changing real world. Specifically, we hope the presented work can contribute to solving the following questions:

- How could we design methods that are resilient to repeatedly being wrong - e.g. repeated wrong loop closures, wrong relocalization after kidnapping?
- How could we design systems that accept that metric localization and perfect scale retrieval are not attainable at all times? What sort of reasoning and planning should agents perform when it is unattainable?
- Is a one map policy viable for lifelong learning? Would a hybrid map-based and sequence-based approach be possible? How can neuroscience findings on memory and cognitive maps help us?
- How should the mechanisms for determining staticity or navigation-usefulness of perceived features be designed? How are they handled in the brain?

REFERENCES

[1] P. Anderson *et al.*, "On Evaluation of Embodied Navigation Agents," *ArXiv*, vol. abs/1807.06757, 2018.

[2] C. Cadena *et al.*, "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age," *IEEE Transactions on Robotics*, vol. 32, pp. 1309–1332, 2016.

[3] S. Lowry *et al.*, "Visual Place Recognition: A Survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.

[4] M. Milford *et al.*, "RatSLAM: a hippocampal model for simultaneous localization and mapping," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 1, 2004, pp. 403–408 Vol.1.

[5] E. Wijmans *et al.*, "Emergence of Maps in the Memories of Blind Navigation Agents," 2023.

[6] K. Ebadi *et al.*, "DARE-SLAM: Degeneracy-Aware and Resilient Loop Closing in Perceptually-Degraded Environments," *Journal of Intelligent & Robotic Systems*, vol. 102, 2021.

[7] E. I. Moser *et al.*, "Place cells, grid cells, and the brain's spatial representation system." *Annual review of neuroscience*, vol. 31, pp. 69–89, 2008.

[8] R. Epstein *et al.*, "The cognitive map in humans: Spatial navigation and beyond," *Nature Neuroscience*, vol. 20, pp. 1504–1513, 10 2017.

[9] M. T. Banich *et al.*, *Cognitive Neuroscience*, 4th ed. Cambridge University Press, 2018.

[10] Y. Song *et al.*, "Flightmare: A Flexible Quadrotor Simulator," in *Proceedings of the 2020 Conference on Robot Learning*, 2021, pp. 1147–1157.

[11] A. Dosovitskiy *et al.*, "CARLA: An Open Urban Driving Simulator," *ArXiv*, vol. abs/1711.03938, 2017.

[12] S. Shah *et al.*, "AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles," in *International Symposium on Field and Service Robotics*, 2017.

[13] A. Szot *et al.*, "Habitat 2.0: Training Home Assistants to Rearrange their Habitat," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[14] M. Deitke *et al.*, "RoboTHOR: An Open Simulation-to-Real Embodied AI Platform," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3161–3171, 2020.

[15] P. Anderson *et al.*, "Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments," in *CVPR*, 06 2018, pp. 3674–3683.

[16] M. Tranzatto *et al.*, "Team CERBERUS Wins the DARPA Subterranean Challenge: Technical Overview and Lessons Learned," *ArXiv*, vol. abs/2207.04914, 2022.

[17] P. Chattopadhyay *et al.*, "RobustNav: Towards Benchmarking Robustness in Embodied Navigation," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15 671–15 680, 2021.

[18] M. Deitke *et al.*, "ProcTHOR: Large-Scale Embodied AI Using Procedural Generation," *ArXiv*, vol. abs/2206.06994, 2022.

[19] M. Kasper *et al.*, "A Benchmark for Visual-Inertial Odometry Systems Employing Onboard Illumination," in *Intelligent Robots and Systems (IROS)*, 2019.

[20] P. Wenzel *et al.*, "4Seasons: A Cross-Season Dataset for Multi-Weather SLAM in Autonomous Driving," in *Proceedings of the German Conference on Pattern Recognition (GCPR)*, 2020.

[21] J. Platt *et al.*, "Comparative Analysis of ROS-Unity3D and ROS-Gazebo for Mobile Ground Robot Simulation," *Journal of Intelligent & Robotic Systems*, vol. 106, 12 2022.

[22] G. Brockman *et al.*, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.