# Arena-Bench: A Benchmarking Suite for Obstacle Avoidance Approaches in Highly Dynamic Environments

Linh Kästner[1], Teham Bhuiyan[1], Tuan Anh Le[1], Elias Treis[1], Johannes Cox[1], Boris Meinardus[1], Jacek Kmiecik[1], Reyk Carstens[1], Duc Pichel[1], Bassel Fatloun[1], Niloufar Khorsandi[1] and Jens Lambrecht[1]

***Abstract*—In recent years, DRL approaches have shown superior performance in dynamic obstacle avoidance. However, these learning-based approaches are often developed in specially designed simulation environments and are hard to test against conventional planning approaches. Furthermore, the integration and deployment of these approaches into real robotic platforms are not yet completely solved. In this paper, we present Arena-bench, a benchmark suite to train, test, and evaluate navigation planners on different robotic platforms within 3D environments. It provides tools to design and generate highly dynamic evaluation worlds, scenarios, and tasks for autonomous navigation and is fully integrated into the robot operating system. To demonstrate the functionalities of our suite, we trained a DRL agent on our platform and compared it against a variety of existing different model-based and learning-based navigation approaches on a variety of relevant metrics. Finally, we deployed the approaches towards real robots and demonstrated the reproducibility of the results. The code is publicly available at github.com/ignc-research/arena-bench.**

## I. Introduction

In recent years, Deep Reinforcement Learning (DRL) has accomplished remarkable results for dynamic obstacle avoidance due to its ability to swiftly react to unexpected changes [1],[2],[3]. A common barrier is that most of the research work evaluated their approaches on specifically designed simulation environments or test setups, making a general comparison against existing approaches difficult [4]. Furthermore, deployment and integration of DRL into real robotic platforms is still an open frontier due to safety reasons [5], [6]. Thus, a benchmark to properly assess those approaches in realistic scenarios and against existing algorithms is not only an essential step towards the deployment of DRL into real robots but also assists in the development and validation of learning-based approaches on mobile robots. Existing benchmarks for robot navigation algorithms mostly focus on static environments, but few exist that cover both dynamic and static ones. Moreover, existing benchmarks for navigation in dynamic environments often contain and compare only a small number of planners [7], [8], [9]. On that account, we propose Arena-bench, a benchmark suite consisting of tools to train, test, and evaluate navigation algorithms for dynamic obstacle avoidance on different robotic systems. This benchmark provides an intuitive interface to design and create dynamic scenarios within 2D and 3D simulators based on Flatland and Gazebo, respectively. The benchmark is completely integrated into the robot operating system (ROS) and includes a variety of classic and state-of-the-art learning-based planners. Arena-bench further provides tools to evaluate all planners in terms of various navigational metrics ranging from navigational safety and robustness to path quality and efficiency.

[1]Chair Industry Grade Networks and Clouds, Faculty of Electrical Engineering, and Computer Science, Berlin Institute of Technology, Berlin, Germany d.kaestner@tu-berlin.de

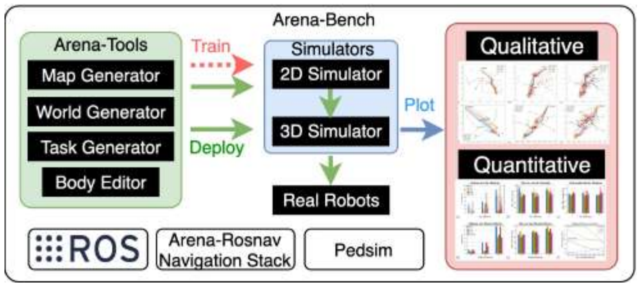


Fig. 1: Arena-bench is a benchmark suite that enables to train and evaluate navigation approaches in realistic dynamic environments. It provides tools to develop navigation approaches, design and generate scenarios, and evaluation tasks on a variety of robotic platforms. The results can be plotted on up to 16 different navigational metrics.

The main contributions of this work are the following:

- Proposal of a benchmark suite to develop navigation approaches and design and generate realistic, dynamic scenarios. The Pedsim library is utilized to realistically model dynamic obstacles.
- Integration of a complete training pipeline to train DRL agents on different robots. The user can integrate new robots and train DRL algorithms on an efficient 2D simulator or on a more realistic 3D simulator.
- Extensive evaluation of several planners developed using this work and provision of tools to evaluate the results on up to 16 relevant navigational metrics.

## II. Methodology

Our proposed benchmark consists of multiple modules that enable the user to design and generate evaluation scenarios, train DRL-based navigation algorithms, integrate ROS-based planners, and plot the results with up to 16 different navigational metrics.

### A. System Design

The complete system design is illustrated in Figure 2. The first module in the benchmark suite is called arena-tools, which includes an extensive toolset to design and generate maps, scenarios, and tasks. The specific components are described in the next section. The user can choose between a manual task generation mode consisting of generating their

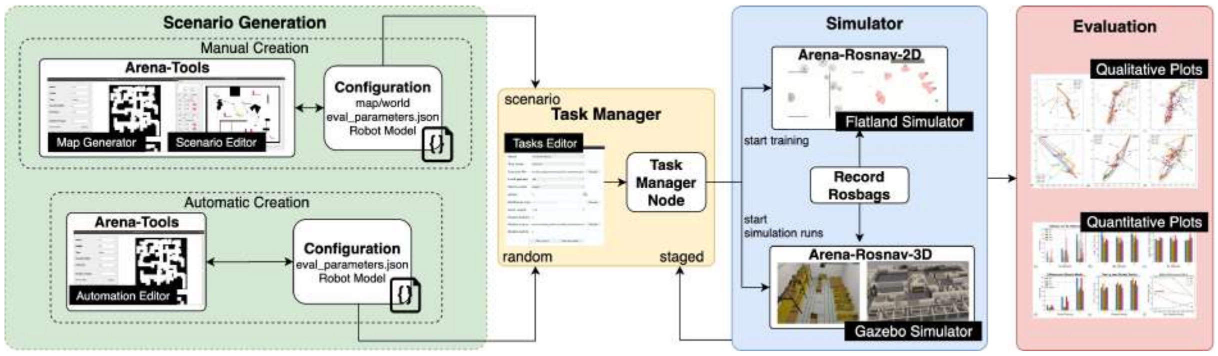


Fig. 2: Our proposed benchmark suite consists of multiple modules for designing and generating different scenarios and evaluating different navigation approaches on specifically designed tasks.

own map, specific scenarios, and an automatic generation of random tasks. Using these tasks, the navigation approaches can be tested and benchmarked using the simulator module. Our framework is built on top of the 2D simulator Flatland and the 3D simulator Gazebo. Moreover, we provide the possibility to train DRL agents in both simulation environments. The resulting planners are cross-compatible across both simulators and utilize ROS as the middleware. Finally, Arena-bench provides an automatic evaluation class to qualitatively and quantitatively evaluate all planning algorithms on up to 14 different navigation metrics ranging from navigational safety to trajectory quality and efficiency.

## *B. Dynamic Obstacles with Pedsim*

The dynamic obstacles are spawned and controlled using this Pedsim library which, includes social states and calculates trajectories [10]. We integrated Pedsim into our 2D and 3D simulator for more realistic behavior.

## *C. Map Generator*

Part of arena-tools is the map editor, which generates 2D and 3D worlds. The user can either manually create a scenario with dynamic and static obstacles or select parameters for automatic scenario generation. Both can be done with our arena-tools application, which provides a variety of functionalities for scenario generation. Thus, even specific scenarios from the end-user can be constructed, which is crucial for real-world validations. Exemplary maps are illustrated in Fig. 4. We integrated the random map generation approach of Heiden et al. [11] and extended it for generating 3D gazebo environments.

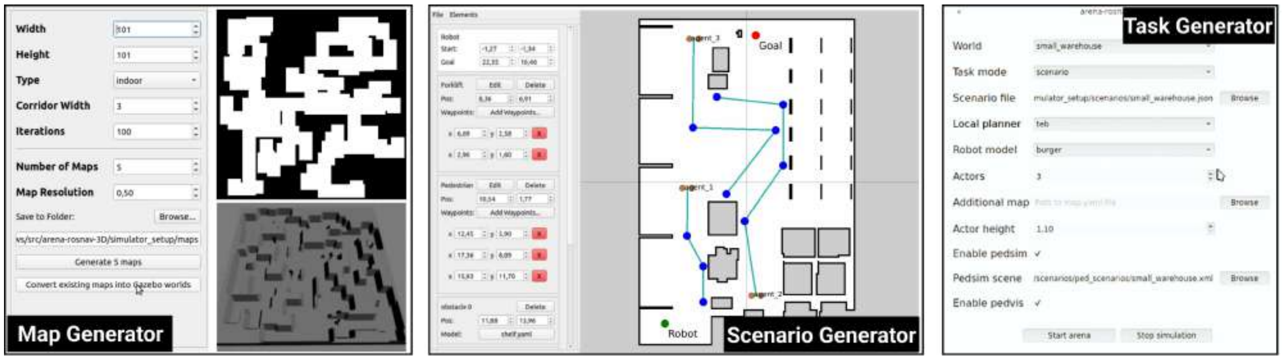
## *D. Task Generator*

Once scenarios are generated, tasks within these scenarios can be defined using the task editor. Thereby, the user defines configuration files, which indicate the task mode, the number of runs, used planners, and the robot model. Fig. 3 illustrates the editor with all parameters to set. In total, the task generator has three modes: Random, Scenario, and Staged, which will be described in the following sections.

**The Random Mode** creates a random scenario for the used Gazebo world in each run. Different arbitrary locations are used for spawning the robot, as well as setting its desired goal position. The number of dynamic obstacles is specified before starting the simulation and fixed for the whole duration. The actors' starting and goal position is randomly generated based on the provided map. This information is then sent to the Pedsim simulator when creating the desired Pedsim obstacles. The randomness of this mode makes it the preferred choice for quantitatively benchmarking different navigation approaches assuming that a large number of random scenarios are created and evaluated to ensure statistically significant evaluations.

**The Scenario Mode** is a reasonable choice for both qualitative and quantitative evaluations. The idea is to limit the randomness without sacrificing complexity to deliver reproducible scenarios, which can be tested and evaluated using separate planning methods. Consequently, each run of this task mode should provide the same environment except for the dynamic behavior of the robot and human agents, which can not be accounted for. We achieve this through arena-tools [12], where a scenario editor was developed. Existing maps of the environment can be used to then specify the robot's spawning location and goal position. Furthermore, we can add an arbitrary number of dynamic obstacles, describe their start/goal spot, as well as their waypoints, which should be included in their overall trajectory. The scenario editor generates a configuration file describing the specifics of a scenario. The data is read by the task_manager, converted into a compatible format for our platform, and subsequently used in the simulation.

**Staged Mode** The staged mode is the preferred choice to automate a long list of evaluation scenarios. Here, the user specifies several levels with increasing difficulty and can set the threshold of when to reach this level. Thus, an evaluation curriculum is generated, which makes it an appropriate choice, especially for quantitative evaluations of planners. The stages can also be designed using arena-tools and are encoded into a configuration file.



Width 101
Height 101
Type indoor
Corridor Width 3
Iterations 100
Number of Maps 5
Map Resolution 0,50
Save to Folder: ws/src/arena-rosnav-3D/simulator_setup/maps

Robot
Start: -1,27 -1,34
Goal 22,33 16,40
Forklift
Pos: 8,36 6,91
Waypoints:
x 6,69 y 2,58
x 2,96 y 1,60
Pedestrian
Pos: 10,54 1,77
Waypoints:
x 12,45 y 3,90
x 17,39 y 6,09
x 15,83 y 11,70
obstacle 0
Pos: 11,88 13,96
Model: shelf.yaml

World small_warehouse
Task mode scenario
Scenario file mulator_setup/scenarios/small_warehouse.json
Local planner teb
Robot model burger
Actors 3
Additional map
Actor height 1.10
Enable pedsim ✓
Pedsim scene /scenarios/ped_scenarios/small_warehouse.xml
Enable pedvis ✓

Fig. 3: Arena-tools. We provide a variety of tools within our benchmark to generate or load existing maps, scenarios, and tasks. The detailed documentation of arena-tools can be found in the GitHub documentation.

## III. Evaluation

To demonstrate the functionalities of our benchmark, we conducted experiments on several different maps. Each map contains two scenarios with 5 and 10 pedestrians with an average speed of 0.3m/s. In each scenario, the obstacle velocities are set to 0.3 m/s. As a global planner, A-Star is used for all approaches. For the DRL-based planners, we utilized the Spatial Horizon waypoint generator of Kästner et al. [13] using the time and location horizon of $t_{lim} = 4s$ and $d_{ahead} = 2m$. For the model-based approaches, we utilize the ROS move-base interface. Localization is acquired using the Adaptive Monte Carlo (AMCL) module. For each planner, we conduct 15 test runs on each scenario. During the test runs, our platform will record the necessary data automatically and provide it to our evaluation class, which is able to generate qualitative and quantitative plots on up to 16 metrics listed in Table **??**. It includes metrics to evaluate navigational efficiency, robustness, safety, and smoothness. Furthermore, some metrics are subdivided in sub-metrics such as the average, minimum, maximum and normalized values, which could give additional insight.

To represent navigational safety, efficiency, and smoothness, the aggregated collision rates, path lengths in meters, and movement jerks of all planners for three different robot platforms on each map are depicted in Fig. 4 respectively. Thereby, a collision is counted when the laser scan detects a value smaller than the robot radius. The movement jerk expresses the rate at which the robot changes its acceleration with respect to time. To represent all common robotic kinematic platforms, we deployed all planners except for the GRING planner on the Jackal (Ackerman Drive), the Turtlebot3 (TB3) (Differential Drive), and the Robotino (RTO) (Holonomic) robot. The GRING planner was only deployed on the Jackal and Turtlebot3 because it was only trained for wheeled robots, and deployment on the Robotino resulted in flawed behavior.

**Navigational Safety:** It is observed that the AIO planner excels in terms of navigational safety, accomplishing the lowest collision rates in all scenarios with 5 and 10 obstacles. Our ROSNAV planner also accomplishes competitive results in terms of navigational safety with low collision rates. Only for the Robotino, ROSNAV produced collisions in the indoor and outdoor map. On all other robot platforms and maps, no collisions were produced. Based on the presented results, the classic move-base planners TEB, DWA, and MPC planners follow up in terms of navigational safety with competitive results over all robots and maps. In general, they also produce low collisions and high success rates over all scenarios and on all robots. However, only DWA on the Turlebot3 produces high collision rates. Aggravating factors are the slow maximum velocity of the Turlebot3 and computationally more demanding calculation times for DWA, which result in slower reaction times to incoming obstacles. The performance of the other learning-based approaches GRING and NAVREP is competitive in scenarios with 5 obstacles but drops significantly for scenarios with 10 obstacles. In situations with a high amount of dynamic obstacles, both planners are not able to react in time.

**Navigational Efficiency:** With regards to navigation efficiency, differences between the robot platforms can be observed. The AIO planner produces high path lengths for the Jackal and Turlebot3. However, for the Robotino, AIO is significantly more efficient, accomplishing one of the lowest path lengths compared to all planners. Whereas ROSNAV results are mediocre for the Jackal and Turlebot3, it outperforms all other planners on the Robotino, similar to our AIO planner. This indicates that our ROSNAV planner works best on holonomic platforms due to the more flexible set of movements the robot can exercise. The move-base planners TEB and DWA produce similar results with competitive results over all robots, maps, and scenarios. However, the MPC planner produces competitive results only on the Jackal and Turlebot3 but performs worst on the holonomic Robotino. Since MPC was specifically designed for car-like robots, these results are expected. GRING and NAVREP perform worst with regards to efficiency producing high path lengths in most scenarios and all robots.

**Navigational Smoothness:** In terms of navigational smoothness, ROSNAV produces mediocre results with higher movement jerk values compared to the classic model-based planners DWA, MPC, and TEB, which all produce competitive results. Especially the MPC planner accomplishes the lowest

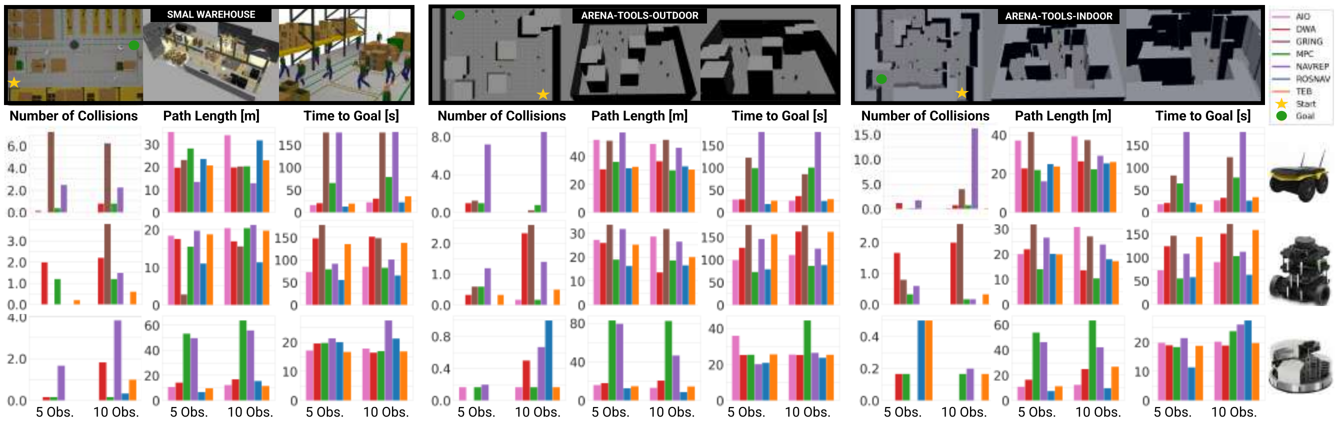


Fig. 4: Experiments in simulation. Quantitative results of all planners on three robots over three worlds within the 3D simulation over the number of obstacles.

movement jerk values on all robots and scenarios. Whereas on the Jackal and Robotino robots, the movement jerk of MPC is close to zero, the jerk is higher on the Turlebot3. Similar observations can be made for most of the other planners, which produce higher jerks for the Turlebot3. This is due to the low maximum velocity of the robot, which results in abrupt velocity changes that cause higher values. The findings also show that the movement jerk is directly correlated with the maximum velocity of the robot. Thus, the Robotino produces a significantly higher jerk compared to the Turlebot3 robot. Nevertheless, the MPC planner is still able to accomplish a very low movement jerk on the Robotino and outperforms all other planners in terms of navigational smoothness.

## IV. Conclusion

In this paper, we proposed Arena-bench, a benchmarking suite to train, test, and compare navigation approaches within ROS in highly dynamic simulation- and real-world environments. Therefore, we introduced a set of tools to design and generate worlds, scenarios, and tasks. Our platform enables the development and training of DRL approaches as well as the integration of classic planners for different robots. To demonstrate the functionalities, we included three different robot types, trained DRL agents on them, and included three conventional planners. Subsequently, we created a variety of test scenarios and conducted extensive evaluations of all approaches with a wide range of alternative navigation approaches. Finally, we transfer the approaches towards real robots and conduct field experiments to assess the sim-to-real gap. The platform provides an important tool not only to assist researchers in developing and evaluating navigation approaches in dynamic environments but also in deploying DRL-based planners towards real robots. In future works, we aspire to do more extensive evaluations on real robotic systems and compare the impact of different robot kinematics on the performance of the navigation approaches. Furthermore, we aim to elucidate the impact of different simulation environments for training and testing.

## References

[1] H.-T. L. Chiang, A. Faust, M. Fiser, and A. Francis, "Learning navigation behaviors end-to-end with autorl," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2007–2014, 2019.

[2] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1343–1350.

[3] A. Faust, K. Oslund, O. Ramirez, A. Francis, L. Tapia, M. Fiser, and J. Davidson, "Prm-rl: Long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5113–5120.

[4] D. Dugas, J. Nieto, R. Siegwart, and J. J. Chung, "Navrep: Unsupervised representations for reinforcement learning of robot navigation in dynamic human environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 7829–7835.

[5] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[6] Z. Zhu and H. Zhao, "A survey of deep rl and il for autonomous driving policy learning," *arXiv preprint arXiv:2101.01993*, 2021.

[7] J. Wen, X. Zhang, Q. Bi, Z. Pan, Y. Feng, J. Yuan, and Y. Fang, "Mrpb 1.0: A unified benchmark for the evaluation of mobile robot local planning approaches," *arXiv preprint arXiv:2011.00491*, 2020.

[8] J. Weisz, Y. Huang, F. Lier, S. Sethumadhavan, and P. Allen, "Robobench: Towards sustainable robotics system benchmarking," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 3383–3389.

[9] I. Rañó and J. Minguez, "Steps towards the automatic evaluation of robot obstacle avoidance algorithms," in *Proc. of workshop of benchmarking in robotics, in the IEEE/RSJ int. conf. on intelligent robots and systems (IROS)*, vol. 88, 2006, pp. 90–91.

[10] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[11] E. Heiden, L. Palmieri, L. Bruns, K. O. Arras, G. S. Sukhatme, and S. Koenig, "Bench-mr: A motion planning benchmark for wheeled mobile robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4536–4543, 2021.

[12] (2021) Arena-tools. [Online]. Available: https://github.com/ignc-research/arena-tools

[13] L. Kästner, X. Zhao, T. Buiyan, J. Li, Z. Shen, J. Lambrecht, and C. Marx, "Connecting deep-reinforcement-learning-based obstacle avoidance with conventional global planners using waypoint generators," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 1213–1220.