CrowdSim++: Unifying Crowd Navigation and Obstacle Avoidance

Marco Rosano[1], Danilo Leocata[1], Antonino Furnari[1,2] and Giovanni Maria Farinella[1,2]
[1]*FPV@IPLAB, Department of Mathematics and Computer Science, University of Catania, Catania, Italy*
[2]*Next Vision s.r.l., Catania, Italy*

Keywords: Crowd Navigation, Reinforcement Learning, Obstacle Avoidance, Crowd Simulator.

Abstract: In recent years, significant advancements in learning-based technologies have propelled the development of autonomous robotic systems designed to assist humans in challenging scenarios during their daily activities. This research focuses on enhancing robotic perception and control, particularly in navigating complex, crowded environments. Traditional approaches often treat static and dynamic components separately, limiting the robots' real-world performance. We propose CrowdSim++, an extension of the open-source CrowdSim simulator (Chen et al., 2019), to unify crowd navigation and obstacle avoidance. CrowdSim++ enables training navigation policies in dynamically generated environments or real-world floor plans, using a 2D lidar sensor and a "person sensor" for enhanced perception. Our experiments demonstrate that Reinforcement Learning-based navigation policies trained in complex environments with humans outperform those trained in simpler scenarios. Additionally, providing robots with specialized sensors to accurately distinguish between static and dynamic obstacles is essential for achieving superior performance. To advance research in autonomous navigation, the source code and dataset of realistic floor plans are available at the following link.

1 INTRODUCTION

In recent years, there has been significant progress in developing learning-based technologies for robotic systems, aiming to create autonomous agents capable of assisting humans in their daily activities (Chen et al., 2013; Miller, 2006). Research in this field has particularly focused on various sub-problems related to the development of complex robotic agents, including perception for action through algorithms processing data acquired via cameras and other sensors (Chen et al., 2021), mechanical arm control (Dong and Zhang, 2023), and navigation (Otte, 2015). Notably, considerable efforts have been made to combine perception and control to perform complex operations across different environments (Yaar et al., 2024).

Several learning-based solutions have emerged to address the autonomous navigation problem, enabling robots to plan and execute routes to destinations without relying on environmental maps (Savva et al., 2019) or even amidst moving people in crowded settings (Chen et al., 2019). These navigation models are typically trained in simulated environments (Otte, 2015), providing a wide variety of navigation scenarios essential for learning optimal navigation policies, avoiding the difficulties of learning from real observations. Indeed, collecting this experience in the real world would be impractical due to the time required and the fragility and cost of the robot's hardware (Rosano et al., 2023).

Navigating crowded environments is of particular interest as it represents a fundamental requirement for designing robotic systems that operate alongside humans in real environments (Möller et al., 2021). Despite the significant progress in autonomous navigation, the design of systems for crowded environments has often approached the dynamic component (moving people) separately from the static component (environment layout) (Chen et al., 2017).

While previous works have addressed point-goal robot navigation and crowded navigation as separate tasks (Mavrogiannis et al., 2023), we postulate that a robot can benefit from learning to perform both tasks simultaneously. This approach leverages the interactions between humans and the environment (e.g., humans won't collide with walls), leading to more effective navigation strategies. Specifically, we propose to train a navigation policy in simulated environments that mirror the complexity of real-world settings, providing the perceptual capabilities necessary to address the navigation challenges. To support our investigation, we propose CrowdSim++, which ex-

Rosano, M., Leocata, D., Furnari, A. and Farinella, G. M.
CrowdSim++: Unifying Crowd Navigation and Obstacle Avoidance.
DOI: 10.5220/0013192300003890
In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 2*, pages 533-542
ISBN: 978-989-758-737-5; ISSN: 2184-433X

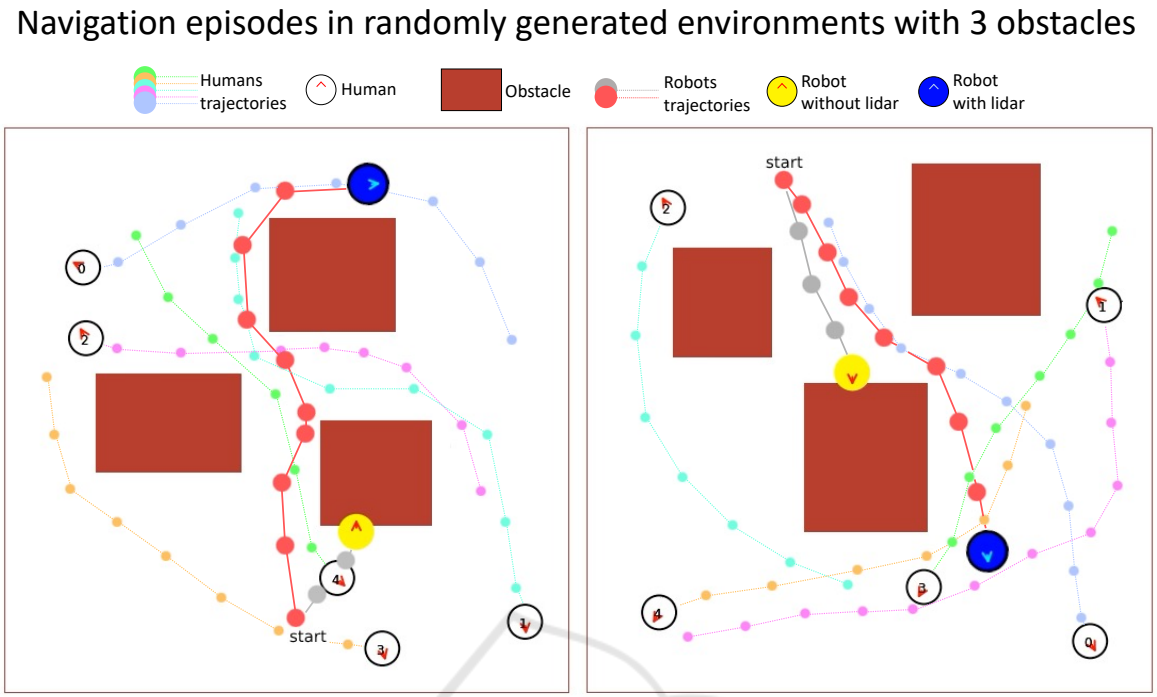

Figure 1: The proposed CrowdSim++ simulator enables the training of policies for crowd navigation in complex environments with both obstacles and moving humans. The robot equipped only with the "person sensor" (yellow circle, gray trajectory) lacks awareness of the environment's layout and exhibits limited performance, even when trained in contexts similar to the test scenario. In contrast, the robot equipped with both the "person sensor" and lidar sensor (blue circle, red trajectory) successfully avoids collisions with static and dynamic obstacles and reaches the goal effectively. The figure represents snapshots of the concluded navigation episodes.

tends the functionality of the CrowdSim open-source simulator (Chen et al., 2019) to set crowd navigation in randomly generated complex environments or in environments derived from real-world floor plans. In the first case, the environments are generated with a variable number of obstacles of different sizes to create scenarios with various levels of complexity. In the second case, we reproduce floor plans from real environments using the popular Gibson dataset (Xia et al., 2018), enabling realistic navigation scenarios. To enable the robot's perception of its surroundings, we implemented a 2D lidar sensor that provides information on the distance of obstacles in the environment. This setup mirrors realistic scenarios where a robot is equipped with lidar to support navigation. Figure 1 shows examples of navigation episodes performed using the proposed simulator.

To validate our claims and show the usefulness of CrowdSim++, we train and evaluate different navigation policies based on Reinforcement Learning (Chen et al., 2019; Chen et al., 2017) in various complex scenarios and compare their performance with the same baselines trained in simpler environments. Our results demonstrate that navigation models significantly benefit from training in more challenging settings. Additionally, equipping robots with proper perception abilities that can clearly observe and differentiate between static and dynamic obstacles is crucial for achieving superior performance.

The contributions of this work are as follows:

- we propose CrowdSim++, an extended version of the CrowdSim simulator (Chen et al., 2019) for training learning-based navigation models in crowded environments with complex geometry. The simulator supports the dynamic generation of environments or the import of floor plans from real environments and, in addition to the "human sensor" originally included in CrowdSim (Chen et al., 2019), includes the use of a lidar sensor, necessary for the robot to perceive its surroundings;
- we demonstrate that navigation policies trained in simpler environments exhibit limited generalization ability when applied to more complex layouts. Through experiments, we show that training navigation models in complex environments with moving humans is essential for developing effective navigation policies.

- we publicly release the simulator's source code along with a floor plan dataset of real-world environments to facilitate the development of advanced navigation systems for complex, crowd-filled scenarios.

2 RELATED WORK

Crowd Navigation Systems. In the context of autonomous navigation in crowded environments, various systems have been proposed over the years to manage the dynamic nature of space configurations. A common approach involves considering pedestrians with simple kinematics, moving at a constant speed along mostly straight trajectories. Reactive methods adopt this assumption and make decisions based on one-step interaction rules to avoid collisions in the immediate future (Van den Berg et al., 2008). However, they fail to model human behavior or the future evolution of the environment, often resulting in unnatural behaviors and the "freezing robot" problem (Pérez-D'Arpino et al., 2021), where the planner halts and repeatedly replans the actions to be taken. To address these limitations, other methods aim to understand human intentions in terms of trajectory and destination, to capture a long-term vision of the environment's evolution (Kuderer et al., 2012).

Recent advancements in deep learning have significantly impacted the field of robot navigation (Otte, 2015). This new approach facilitates learning navigation policies from demonstrations of desired behavior in an end-to-end manner, starting from the robot's raw observations. Imitation Learning and Behavior Cloning strategies leverage expert supervision to train deep models in a supervised manner (Bojarski et al., 2016). Nevertheless, these systems often struggle to generalize beyond the scenarios encountered during training. To overcome the problem, new systems based on Reinforcement Learning (RL) have emerged, capable of autonomously collecting the necessary experience within simulated environments (Rosano et al., 2021; Zhu and Zhang, 2021). These systems have proven effective for learning from large amounts of simulated data (Savva et al., 2019). Nevertheless, these methods are often trained in static environments without moving people. Concurrent research has focused on developing methods for navigation in crowded environments, utilizing simulation systems that can mimic human movement, albeit with simplistic dynamics (Pérez-D'Arpino et al., 2021). For instance, Chen et al. (Chen et al., 2017) proposed an RL-based model that encodes the estimated time to the goal given the positions and velocities of neighbors. Meanwhile, the authors of (Chen et al., 2019) introduced a method to capture crowd-robot interactions in the presence of many humans using a self-attention system to estimate their influence on future states. Unfortunately, most of these approaches involve humans isolated from the environmental context, and only a limited number of methods have addressed the problem using environments with more complex geometries (Liu et al., 2020). In this work, we demonstrate that navigation policies trained in simpler struggle when applied to more complex layouts, and that training navigation models in realistic contexts—featuring both static and dynamic obstacles—can be advantageous for learning advanced navigation policies, leveraging the capabilities of the proposed simulator.

Simulators and Datasets for Navigation Systems. Simulators are fundamental tools for developing robotic navigation systems. Historically, due to the scarcity of advanced simulation platforms, many researchers used simple, ad-hoc simulators (Chen et al., 2017) or commercial products (Staranowicz and Mariottini, 2011). More recently, advanced tools capable of simulating human presence and human-robot interactions have been developed and made freely available to the scientific community (Möller et al., 2021). The usability of these simulators has been greatly enhanced by their integration with robotic middleware that provides a unified environment for using various robot software (Quigley et al., 2009) . The rise of deep learning systems and the success of Reinforcement Learning methods in robotics have spurred a rapid adoption of simulation systems capable of handling large-scale datasets (Zhu and Zhang, 2021). One area of focus has been the development of photorealistic simulators (Savva et al., 2019; Xia et al., 2018), which utilize manually reconstructed indoor environments (Khanna et al., 2024) or those captured using 3D scanners (Xia et al., 2018). In this context, the fidelity of virtual images-ensuring they accurately reflect the characteristics of real-world images-is crucial for applying the navigation system in real environments. Concurrently, efforts have concentrated on simulators designed for crowd navigation, capable of realistically simulating human movement within reconstructed environments (Chen et al., 2019; Kuderer et al., 2012). These simulators typically employ high-level sensors, thus they focused more on emulating human movement than on visual rendering (Van den Berg et al., 2008; Chen et al., 2019). However, only few studies have explored joint scenarios involving navigation in complex environments alongside human presence (Liu et al., 2020). In contrast to previous studies (Cancelli et al., 2023; Pérez-D'Arpino et al.,

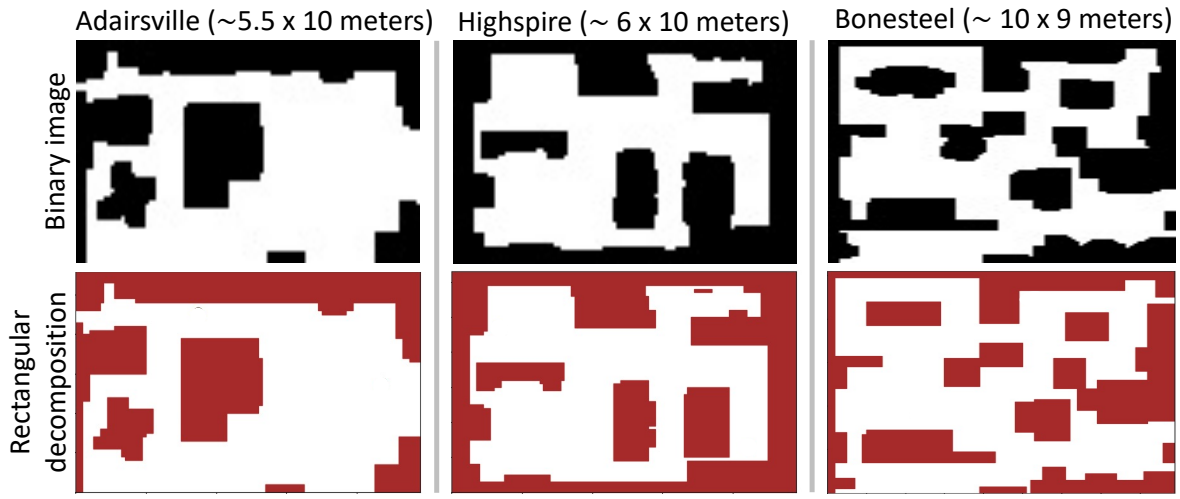

Figure 2: Examples of floor plans of real environments from the Gibson dataset (Xia et al., 2018). After extracting the binary images of the navigable areas (first row), each floor plan was decomposed into a set of rectangles (second row). This representation simplifies the process of importing floor plans into the proposed simulator. The floor plans have been scaled for visualization purposes, their actual size is indicated in the text above each image.

2021), which employed unrealistic human dynamics, this work extends the simulator introduced in (Chen et al., 2019) to support advanced human simulations for recreating complex navigation environments. The simulator allows dynamic environment generation or the import of real floor plans and incorporates a lidar sensor for obstacle perception. Similar to the "person sensor", the signal from the lidar is less affected by the potential domain gap between the simulated and real world. Navigation models trained under these conditions are better equipped to operate in realistic settings, facilitating the learning of robust and sophisticated navigation policies.

3 TRAIN NAVIGATION MODELS IN COMPLEX ENVIRONMENTS

In this section, we provide a detailed discussion on the proposed simulator designed for training navigation policies in environments that include both static components (such as the environment's layout) and dynamic components (such as moving humans). We will release the code of the proposed simulator, along with the dataset of 2D floor plans from real-world environments.

3.1 The CrowdSim++ Simulator

The proposed CrowdSim++ is an enhanced version of the open-source simulator introduced in (Chen et al., 2019). This simulator represents the environment on a 2D plane, viewed from a top-down perspective. Figure 1 displays two examples of navigation episodes performed within the simulator. Both the robot and humans are depicted as circles, moving freely on the 2D plane and guided toward their destinations by the respective navigation policies. The simulator natively implements the Gym library, enabling the use of numerous Gym-compliant policy optimization algorithms available in the state of the art. The layouts of the environments used in simulation were recreated using rectangles of varying dimensions. When combined, these rectangles form more complex geometries. Examples of the generated environments are depicted in Figure 2. The proposed system allows for the creation of two distinct types of scenarios: 1) randomly generated environments with a variable number of obstacles; 2) environments recreated based on the floor plans of real locations. The construction methods for these scenarios are described below.

Randomly Generated Environments. This strategy offers a practical solution for constructing synthetic environments with varying geometry and complexity, crucial for training robust navigation policies capable of generalizing to novel environments. Example of such environments are showed in Figure 1. The selection of obstacle count and size allows for tailoring the complexity level encountered by the navigation model during both training and testing phases. During environment generation, obstacles are strategically placed to maintain a margin from the environment boundaries and each other. This approach ensures that invalid configurations, such as

edge-concentrated (simple layout) or obstructed environments lacking sufficient navigation space (partitioned environment), are avoided. This study explores various types of randomly generated environments, progressively increasing in complexity.

Real Environments. The use of floor plans derived from real environments holds significant importance as it allows navigation models to learn policies directly from observations of complex geometries resembling potential deployment environments for robots. Realistic floor plans also prove beneficial in scenarios requiring specialized navigation models tailored to specific environments or when early performance estimates are necessary before real-world testing. Figure 2 displays examples of the floor plans, represented as binary images, along with their respective decomposition into rectangles. In this study, we focused on the 3D indoor environments from the Gibson dataset (Xia et al., 2018), comprising 572 real buildings reconstructed using advanced 3D scanning techniques. Given that our simulator supports 2D floor plans, a preprocessing step was required to adapt these environments. To achieve this, we imported the 3D models into the Habitat simulator (Savva et al., 2019) and developed a custom script to extract the navigation meshes (navmeshes), i.e. polygon meshes defining navigable areas for agents. Optimal results were achieved by setting the virtual agent dimensions to a height of 0.1 meters and a diameter of 0.2 meters. For multi-level buildings, only ground floor rooms were included. Navmesh data was subsequently converted into binary images to differentiate navigable areas (white) from obstacles (black), with each pixel representing 0.1 square meters. To enhance navigability while preserving geometry, morphological operators were applied, notably two iterations of the *closing* operation, yielding satisfactory results. Finally, to import the floor plan into the simulator, we employed a *rectangular decomposition algorithm*[1]. This transformed non-navigable areas of the 2D binary image into a set of variable-sized rectangles. The algorithm minimizes the number of rectangles while maximizing coverage of the specific area. Following generation, the rectangles were sorted based on area size (from largest to smallest). We then set a threshold of 30 as the maximum number of rectangles per floor plan for use during navigation simulations. In our study, we adopted the "medium" data split of the Gibson dataset as proposed in (Xia et al., 2018), which contains a subset of environments of enhanced quality. Throughout the transformation process of these floor plans, some were excluded due to lack of navigability or disconnected areas. The resulting dataset is a curated collection comprising 120 distinct environments.

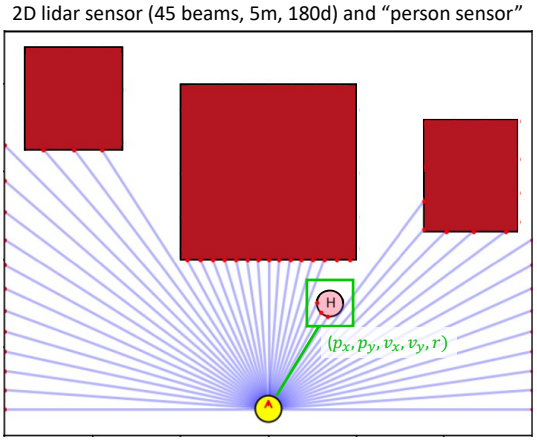

Figure 3: Visualization of the sensors used by the robot. The purple segments originating from the robot (yellow circle) represent the laser beams of the lidar sensor. The red dots on the obstacles indicate the distances perceived by each laser beam. In green, the "person sensor" is highlighted, which provides information on the position, speed, and space occupied by the humans.

Robot's Perception. Effective navigation for a robot relies on having the appropriate sensors to perceive its environment accurately. In this regard, we extended the CrowdSim simulator (Chen et al., 2019) to support a 2D lidar sensor, which allows the robot to get detailed and real-time information about obstacles, walls, and other structural elements of the environment. Figure 3 provides a graphical representation of the lidar sensor. Specifically, it measures the distance to objects by calculating the time it takes for the laser pulses to reflect back from surfaces. The implemented sensor can be customized by adjusting parameters such as the number of laser beams, maximum perceived distance, and field of view (up to 360 degrees). An obstacle, whether static or dynamic, is detected by a laser beam if there is an intersection between the beam and the obstacle. In such cases, the measured distance is that of the closest intersection point to the robot. We employed the *Shapely*[2] library for efficient management of geometric intersections. Additionally, we implemented a strategy to enhance measurement efficiency by excluding obstacles beyond the sensor's maximum range.

[1]We used the Adaptive-Boxes decomposition algorithm https://github.com/jnfran92/adaptive-boxes

[2]We used the Shapely library https://github.com/shapely/shapely

4 EXPERIMENTAL SETTINGS AND RESULTS

In this section, we discuss the characteristics of the environments used for training navigation policies, the considered navigation models, and the details of the training and evaluation setup.

4.1 Environments and Simulation

To test our hypothesis that training navigation policies in complex, crowded environments leads to superior performance, we designed a series of navigation environments with varying levels of difficulty.

Generated Environments. For randomly generated environments, we trained and tested the navigation models in scenarios with one and three obstacles. We set each environment to a fixed size of 10×10 meters. Within this space, we generated rectangular obstacles with variable dimensions, with side lengths randomly chosen in the range $[3,6]$ meters. In all cases, we added four additional obstacles along the edges of the environment. This served two purposes: 1) preventing agents from navigating outside the environment without triggering a collision event; 2) enabling the lidar sensor to accurately detect the boundaries of the navigable space. Subsequently, To ensure challenging navigation tasks, we placed agents' starting positions and goals at least 7 meters apart (70% of the environment's longest side). To enhance simulation efficiency, we pre-computed obstacle layouts and agent start/goal positions, loading them as needed instead of generating them in real-time.

For each scenario, we created 100 unique environments for both the training and test sets. Each environment supports thousands of possible navigation episodes. This setup allows the navigation model to learn from a diverse range of similarly complex environments, enhancing its ability to generalize to new settings. The number of episodes per environment is dynamically calculated at the start of each training or testing session, based on the total number of navigation episodes.

Real Environments. Real floor plans provide an opportunity to develop navigation policies for highly complex environments. In our setup, we focused on training specialized models for navigation in specific environments where floor plans are available in advance. This scenario is common in practice, as robots often need in-depth knowledge of environmental layouts to operate with high accuracy. Moreover, this approach aligns with classic navigation systems that require access to environmental maps (Durrant-Whyte and Bailey, 2006). Specifically, we selected three real environments from our proposed set of floor plans, representing *easy*, *medium*, and *hard* scenarios with increasing levels of difficulty. Figure 2 illustrates these floor plans. For each environment, we sampled two disjoint sets of trajectories: one for training and one for testing. As with the random obstacle scenarios, we pre-computed these trajectories to enhance simulation efficiency.

Simulation Details. Our simulations feature environments with varying numbers of human agents during the training phase. For evaluation purposes, we consistently use scenes containing five human agents. All agents, including the robot, are represented as circular entities with a radius of 0.3 meters. Humans are guided by the ORCA policy (Van den Berg et al., 2008). We assume that the robot has holonomic kinematics and that it is invisible to humans. Navigation is formulated as a point-goal navigation task (Möller et al., 2021). At each step, the robot receives a new observation from the environment, updates the information on the goal to be reached and chooses one of 16 discrete actions. The observation includes the robot's self state $\mathbf{w}^t$ and, depending on the type of input considered (as we will see in Section 4.2), the lidar observation $\mathbf{l}^t$, the states of humans $\mathbf{u}^t = [\mathbf{u}_1^t, \mathbf{u}_2^t, ..., \mathbf{u}_n^t]$ assuming a total number of n humans, or both of them. All data are expressed in the local reference frame of the robot. The robot state $\mathbf{w}^t$ consists of the robot's distance to goal d_g, preferred speed v_{pref}, velocity (v_x, v_y), radius r. Each human state $\mathbf{u}_i^t$ consists of the human's position $\left(p_x^i, p_y^i\right)$, velocity $\left(v_x^i, v_y^i\right)$ and radius r^i. The lidar observation $\mathbf{l}^t$ consists of n_l range measurements indicating the distance to the closest obstacle from each beam's direction. The action space consists of 2 linear velocities exponentially spaced between $(0, v_{pref}]\, m/s$ and 8 directions evenly spaced between $[0, 2\pi)$ degrees. The navigation episode ends if the robot reaches the goal within a radius of 0.2 meters, when a collision occurs or when the maximum simulation time is reached. Each step is equivalent to an execution time of $0.25s$ and the maximum simulation time for an episode is $25s$.

4.2 Navigation Models

In our study, we considered two popular Reinforcement Learning-based models for crowd navigation: CADRL (Chen et al., 2017) and SARL (Chen et al., 2019). In their original formulation, both models rely on direct input from a "person sensor" that provides information on the position, velocity, and physical dimensions of nearby humans. SARL represents an improvement over CADRL as it adopts a more ad-

vanced input data aggregation system based on a self-attention system, to better capture the robot's interactions with the most relevant humans. In this work, we modified the CADRL and SARL models to process 2D lidar signals, allowing us to evaluate their performance in complex, crowded environments under various sensor configurations. We conducted experiments to assess the navigation models' capabilities when equipped with: 1) "person sensor" only; 2) lidar sensor only; 3) a combination of "person sensor" and lidar sensor. From now on, we will refer to these navigation models as CADRL++ and SARL++, respectively. For the lidar configuration, we optimized sensor parameters to balance environmental perception and simulation efficiency. We set the Field of View (FOV) to 180 degrees, covering the robot's frontal semicircle. The lidar uses 45 laser beams and has a maximum perceptible distance of 5 meters.

4.3 Training and Evaluation Setup

Following the setup in (Chen et al., 2019), the navigation policies were bootstrapped from the demonstrations provided by ORCA (Van den Berg et al., 2008) using imitation learning, for a total of $3k$ navigation episodes. Subsequently, they were trained using Reinforcement Learning for $10k$ episodes, using the $\varepsilon-$greedy exploration strategy, with ε decaying linearly from 0.9 to 0.5 over the first $5k$ navigation episodes. The discount factor γ and the learning rate were set to 0.9 and 0.001, respectively. We found the reward function defined in (Chen et al., 2019) suitable to award task accomplishments while penalizing collisions or uncomfortable distances. The policy is trained by the temporal-difference method with standard experience replay and fixed target network techniques (Chen et al., 2019; Chen et al., 2017). Please refer to Section 1 of the supplementary material for a more detailed formulation of the Reinforcement Learning framework adopted.

In our evaluation process, we assessed navigation performance across 500 trajectories for each navigation scenario. These trajectories were sampled from test environments obtained as indicated in Section 4.1. Specifically, for the randomly generated environments, we conducted tests under two distinct scenarios. The first scenario involved environments containing a single obstacle, while the second one featured environments with three obstacles. In contrast, our evaluation of real floor plans utilized the same environmental layout, but we employed a separate set of navigation trajectories. All scenarios included five human agents.

Our evaluation employed several key metrics to provide a comprehensive evaluation of the navigation system's performance, safety, efficiency, and social awareness: 1) *success rate* quantifies the average proportion of episodes completed without collisions. It provides insight into the overall effectiveness of the navigation system in safely reaching its destination; 2) *collision rate* measures the average fraction of episodes that concluded with the robot colliding with either humans or obstacles. This metric helps identify the frequency of navigation failures due to physical interactions; 3) *timeout rate* represents the average proportion of episodes that reached the maximum simulation time without completion. It indicates scenarios where the robot was unable to reach its goal within the specified time constraints; 4) *navigation time* calculates the average time taken by successful episodes to reach the goal. This metric allows for a more precise evaluation of the policy's performance when it operates as intended; 5) *danger frequency* calculates the average fraction of navigation steps during which the robot came within 0.2 meters (referred to as the discomfort distance) of a human. This metric assesses the robot's ability to maintain comfortable distances from humans during navigation.

5 EXPERIMENTS

In the following experiments, we aim to demonstrate that existing navigation models can benefit from training in environments with both complex layouts and moving humans, particularly when configured to receive input signals from additional sensors.

Results for Generated Environments. Table 1 presents the results of navigation models tested in environments featuring one obstacle and five humans. Each line indicates the type of navigation model, the used sensors (consistent across both the training and testing phases), the number of obstacles, and whether humans were present during training (we fixed the number of humans to five). As expected, the CADRL++ navigation model shows limited performance when trained in obstacle-free environments (lines 1-3). For the model equipped only with a "person sensor", the information on human behavior was insufficient to infer the presence of obstacles, which remain invisible. The model using only the lidar sensor (line 2) was able to perceive static obstacles and reduce the collision rate compared to the model using only the "person sensor" (line 2 vs. 1, 0.3240 vs. 0.4788 collision rate). However, it failed to accurately detect and avoid humans, as indicated by the higher recorded danger frequency. Moreover, its success rate

Table 1: Performance of the considered navigation models, tested in environments with one obstacle and five humans. The policies were trained in scenarios with a variable number of obstacles (ranging from 0 to 3), with or without humans, and using the "person sensor," the lidar sensor, or both. The results confirm that training in crowded environments with a greater number of obstacles leads to better outcomes. Additionally, using both sensors is essential for optimal performance.

	Test environments: 1 obstacle, 5 humans								
	model	sensor(s)	# of obstacles	humans	success rate (↑)	collision rate (↓)	timeout rate (↓)	nav. time (↓)	danger freq. (↓)
1	CADRL	person	0	✓	0.4550	0.4788	0.0662	10.2282	0.0766
2	CADRL++	lidar	0	✓	0.2560	0.3240	0.4200	17.5719	0.1232
3	CADRL++	lidar+person	0	✓	0.4932	0.2500	0.2568	13.5847	0.0946
4	CADRL++	lidar	1	✗	0.4824	0.1480	0.3696	16.1093	0.0871
5	CADRL++	lidar	3	✗	0.5644	0.1900	0.2456	12.3953	0.1664
6	CADRL++	person	1	✓	0.3124	0.4800	0.2076	12.8515	0.0670
7	CADRL++	person	3	✓	0.0120	0.1720	0.8160	25.0000	0.0471
8	CADRL++	lidar	1	✓	0.5134	0.2176	0.2690	16.0571	0.1104
9	CADRL++	lidar	3	✓	0.6027	0.3024	0.0949	12.6989	0.1708
10	CADRL++	lidar+person	1	✓	0.6741	0.1960	0.1299	12.9387	0.0896
11	CADRL++	lidar+person	3	✓	0.7172	0.1510	0.1318	11.2958	0.1659
12	SARL	person	0	✓	0.4727	0.4312	0.0961	9.7005	0.0183
13	SARL++	lidar+person	0	✓	0.5688	0.2880	0.1432	10.9796	0.0150
14	SARL++	person	1	✓	0.3514	0.3390	0.3096	12.2493	0.0252
15	SARL++	person	3	✓	0.0520	0.0900	0.8580	21.8550	0.0624
16	SARL++	lidar+person	1	✓	0.6934	0.2295	0.0771	10.9218	0.0272
17	SARL++	lidar+person	3	✓	0.7498	0.1560	0.0942	11.5688	0.0651

remains limited because it was not exposed to environments with obstacles during training. Adding a "person sensor" to the existing lidar sensor significantly improved navigation performance (line 3). By explicitly detecting the position and speed of humans, the model was able to reduce both the collision rate and the frequency of invading human comfort zones. Training the models equipped with a lidar sensor in environments with obstacles but without humans (line 4-5) further reduced the collision rate, even if humans are not seen during training, suggesting that the lidar can compensate for the lack of a dedicated "person sensor". Using a more complex environment with three obstacles led to a higher success rate (line 5).

For navigation models trained in the presence of humans and with a variable number of obstacles (lines 6-11), we observe that more complex environments are beneficial when the robot uses appropriate sensors. Specifically, when only the "person sensor" is used, performance is very limited (line 7, CADRL++ with three obstacles). However, when the lidar sensor is used (lines 8-9), performance improves drastically. In the scenario with three obstacles, the navigation model learned to navigate the complex geometries, although it showed some difficulty in managing human dynamics, as indicated by a higher danger frequency. The best-performing CADRL++ navigation models are those equipped with both "person" and lidar sensors (lines 10-11). Among these, the model trained in more complex environments achieved the highest success rate (0.7172). The SARL++ navigation model confirmed its superior ability in capturing human-robot relationships.

When analyzing the performance of SARL++ (lines 12-17), it consistently outperformed CADRL++ across all training scenarios. It achieved the best success rate (0.7498) in the scenario where both the "person sensor" and lidar sensor were used, and the training took place in environments with three obstacles and the presence of humans.

Navigation models tested in environments with three obstacles and five humans achieved performances that mirror the trends observed in Table 1 but they are generally inferior. Please refer to Section 2 of the supplementary material for a more detailed discussion.

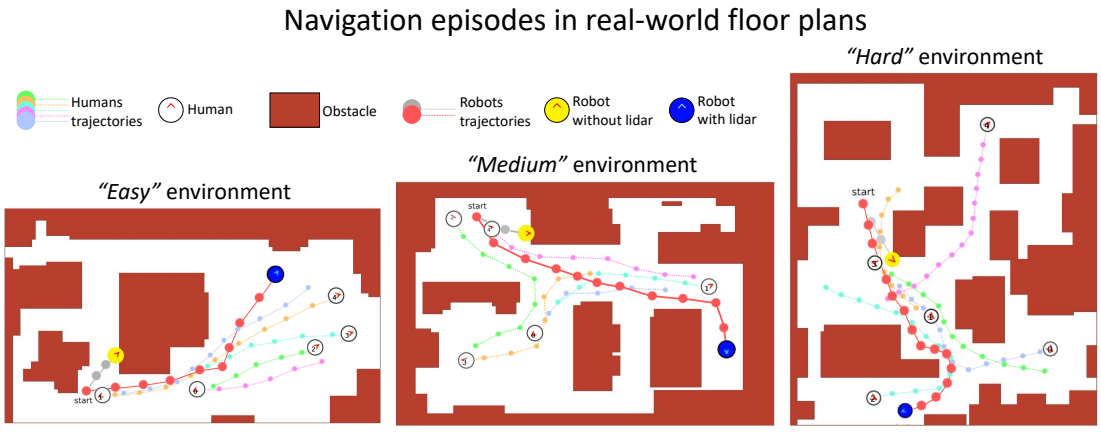

Figure 4: Examples of navigation episodes in real floor plans using the SARL++ model in the *easy*, *medium*, and *hard* scenarios. The robot equipped with both "person" and lidar sensors (blue circle, red trajectory) successfully navigates through narrow and crowded areas. In contrast, using the same navigation model without the lidar sensor, the robot collides with the first obstacle encountered (yellow circle, gray trajectory). The figure represents snapshots of the concluded navigation episodes.

Overall, the results confirm that navigation models can learn advanced navigation policies when trained in complex environments. Training in crowded environments with a greater number of obstacles leads to better results and more robust models. Additionally, the use of specialized sensors is advantageous for capturing the distinct properties of static and dynamic obstacles.

Results for Real Environments. Table 2 reports the results obtained by SARL++ with both "person sensor" and lidar sensor, trained and tested on three real floor plans classified as *easy*, *medium*, and *hard*. The floor plans are showed in Figure 4. This navigation model achieved the best results in the previous evaluations. In the easy scenario, SARL++ performed well, with almost no navigation episodes ending in timeout, indicating that a robust navigation policy had been learned, capable of guiding the robot to its destination efficiently. However, the success rate dropped in the medium and hard scenarios, with the hard scenario showing the highest timeout rate. Similar to the test conducted in environments with three obstacles, this performance decline may be due to the use of a less efficient RL algorithm, which requires a significantly higher number of training episodes to handle more challenging environments effectively.

Nonetheless, the average results confirm that navigation policies benefit from exposure to complex real floor plans and that training specialized models for specific floor plans is an effective strategy.

Table 2: Navigation performance in real-world floor plans. The evaluation focused on the SARL++ model, equipped with both a person sensor and lidar sensor. Refer to the text for details.

model	sensor(s)	real floor plan	humans	success rate	collision rate	timeout rate
SARL++	lidar+person	easy	✓	0.8436	0.1433	0.0131
SARL++	lidar+person	medium	✓	0.7678	0.1842	0.0480
SARL++	lidar+person	hard	✓	0.6610	0.1079	0.2311
avg.				0.7575	0.1451	0.0974

6 CONCLUSION

In this study, we introduced CrowdSim++, an advanced simulator designed to train navigation models under realistic conditions. Through experimentation and comparative analysis, we confirmed the importance of training in complex scenarios involving humans to develop robust navigation policies suitable for real-world deployment. We anticipate that our proposed simulator and dataset of realistic floor plans will catalyze further advancements in autonomous navigation systems.

ACKNOWLEDGEMENTS

This research has been supported by the project Future Artificial Intelligence Research (FAIR) – PNRR MUR Cod. PE0000013 - CUP: E63C22001940006

REFERENCES

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., et al. (2016). End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.

Cancelli, E., Campari, T., Serafini, L., Chang, A. X., and Ballan, L. (2023). Exploiting proximity-aware tasks for embodied social navigation. In *International Conference on Computer Vision (ICCV)*. IEEE.

Chen, B., Sax, A., Lewis, F., Armeni, I., Savarese, S., Zamir, A., Malik, J., and Pinto, L. (2021). Robust policies via mid-level visual representations: An experimental study in manipulation and navigation. In Kober, J., Ramos, F., and Tomlin, C., editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155, pages 2328–2346.

Chen, C., Liu, Y., Kreiss, S., and Alahi, A. (2019). Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning. In *International Conference on Robotics and Automation (ICRA)*.

Chen, T. L., Ciocarlie, M., Cousins, S., Grice, P. M., Hawkins, K., Hsiao, K., Kemp, C. C., King, C.-H., Lazewatsky, D. A., Leeper, A. E., et al. (2013). Robots for humanity: using assistive robotics to empower people with disabilities. *IEEE Robotics & Automation Magazine*, 20(1).

Chen, Y. F., Liu, M., Everett, M., and How, J. P. (2017). Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In *International Conference on Robotics and Automation (ICRA)*.

Dong, M. and Zhang, J. (2023). A review of robotic grasp detection technology. *Robotica*.

Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*.

Khanna, M., Mao, Y., Jiang, H., Haresh, S., Shacklett, B., Batra, D., Clegg, A., Undersander, E., Chang, A. X., and Savva, M. (2024). Habitat synthetic scenes dataset (hssd-200): An analysis of 3d scene scale and realism tradeoffs for objectgoal navigation. In *Computer Vision and Pattern Recognition (CVPR)*.

Kuderer, M., Kretzschmar, H., Sprunk, C., and Burgard, W. (2012). Feature-based prediction of trajectories for socially compliant navigation. In *Robotics: science and systems*, volume 8.

Liu, L., Dugas, D., Cesari, G., Siegwart, R., and Dubé, R. (2020). Robot navigation in crowded environments using deep reinforcement learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Mavrogiannis, C., Baldini, F., Wang, A., Zhao, D., Trautman, P., Steinfeld, A., and Oh, J. (2023). Core challenges of social robot navigation: A survey. *ACM Transactions on Human-Robot Interaction*, 12(3).

Miller, D. P. (2006). Assistive robotics: an overview. *Assistive Technology and Artificial Intelligence: Applications in Robotics, User Interfaces and Natural Language Processing*.

Möller, R., Furnari, A., Battiato, S., Härmä, A., and Farinella, G. M. (2021). A survey on human-aware robot navigation. *Robotics and Autonomous Systems*, 145.

Otte, M. W. (2015). A survey of machine learning approaches to robotic path-planning. *University of Colorado at Boulder, Boulder*.

Pérez-D'Arpino, C., Liu, C., Goebel, P., Martín-Martín, R., and Savarese, S. (2021). Robot navigation in constrained pedestrian environments using reinforcement learning. In *International Conference on Robotics and Automation (ICRA)*.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A. Y., et al. (2009). Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3.

Rosano, M., Furnari, A., Gulino, L., and Farinella, G. M. (2021). On embodied visual navigation in real environments through habitat. In *International Conference on Pattern Recognition (ICPR)*.

Rosano, M., Furnari, A., Gulino, L., Santoro, C., and Farinella, G. M. (2023). Image-based navigation in real-world environments via multiple mid-level representations: Fusion models, benchmark and efficient evaluation. *Autonomous Robots*, 47(8).

Savva, M., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., et al. (2019). Habitat: A platform for embodied ai research. In *International Conference on Computer Vision (ICCV)*.

Staranowicz, A. and Mariottini, G. L. (2011). A survey and comparison of commercial and open-source robotic simulator software. In *Proceedings of the 4th International Conference on PErvasive Technologies Related to Assistive Environments*.

Van den Berg, J., Lin, M., and Manocha, D. (2008). Reciprocal velocity obstacles for real-time multi-agent navigation. In *International Conference on Robotics and Automation (ICRA)*.

Xia, F., Zamir, A. R., He, Z., Sax, A., Malik, J., and Savarese, S. (2018). Gibson env: Real-world perception for embodied agents. In *Computer Vision and Pattern Recognition (CVPR)*.

Yaar, A., Furnari, A., Rosano, M., Härmä, A., and Farinella, G. M. (2024). Finding and navigating to humans in complex environments for assistive task. In *International Conference on Computer Vision Theory and Applications (VISAPP)*.

Zhu, K. and Zhang, T. (2021). Deep reinforcement learning based mobile robot navigation: A review. *Tsinghua Science and Technology*, 26(5).