Toward Real-World Deployment and Long-Term Evaluation of Social Navigation Systems

Nhat Le, Duc M. Nguyen, Ryan Wang, Rohan Bhandange, Alexandyr Card, Xuesu Xiao

***Abstract*— Social navigation research through realistic service-robot tasks requires supporting infrastructure for deployment, task execution, monitoring, and data collection. This paper presents Real-World Deployment (RWD), an open-source ROS 2 framework providing a standardized workflow for deploying robots on user-defined tasks in different buildings. RWD converts service-robot tasks into long-range, potentially multi-floor missions and uses behavior trees to manage task execution with operational monitoring and selective task-associated data logging. Modular interfaces separate reusable components from robot-specific and building-specific integration. The physical evaluation comprises a within-participant visitor-guidance study of three navigation systems and a case study that measures the effort required to deploy RWD on a different robot in a different building. The user study results motivate further testing with RWD, while the deployment case study requires nine hours of active engineering work to deploy the framework for a robot in a large academic building. The source code and deployment resources will be made publicly available upon acceptance.**

I. Introduction

Collecting real-world social navigation data and evaluating navigation algorithms remain costly, difficult to repeat, and often limited to a narrow range of scenarios. Commercial service robots already perform building-scale tasks across diverse environmental and social contexts [1], [2]. However, these deployments generally do not give researchers access to their autonomy stack and task records. Academic evaluations therefore often rely on simulation [3], [4], controlled human-robot interaction scenarios [5]–[9], short point-to-point navigation trials, or teleoperated robot-navigation datasets [10]. These evaluation designs support controlled and repeatable studies, but autonomous service-robot operation requires supporting infrastructure beyond the navigation system itself.

Motivated by the guidelines presented by Francis et al. [11], this paper presents Real-World Deployment (RWD), an open-source ROS 2 framework for deploying navigation methods as part of complete service-robot tasks in researchers' own environments (Fig. 1). RWD supports data collection during extended autonomous operation and navigation-system evaluation through the same task interface and deployment scheme. By executing building-scale tasks across corridors, intersections, doors, and elevators, each deployment exposes a navigation stack to varied spatial and interaction contexts rather than only isolated point-to-point trials. This broader scope supports more holistic evaluation and study of long-term effects, including whether novelty effects diminish as people become familiar with the robot.

RWD targets visitor guidance, package delivery, and loop patrol, for which doors, elevators, and floor transitions are recurring bottlenecks. RWD provides semantic configuration tools, a mission planner to convert each long-range request into navigation and building-interaction steps, and a customizable Behavior Tree (BT)-based task executor to run this task sequence. Reusable ROS 2 packages are separated from robot-specific drivers and sensor settings through templated robot adapters, with a simulation pipeline for a real-to-sim-to-real deployment workflow.



Fig. 1: **Left**: A robot running RWD in its deployment environment. **Middle**: A user's first-person view during visitor guidance. **Right**: The robot leaving an elevator after a ride.

Beyond supporting deployment, RWD provides an experimental boundary for navigation research. Nav2 provides a standardized interface between task execution and navigation, and methods developed outside Nav2 can be adapted to the same interfaces. The common boundary keeps the surrounding infrastructure fixed while allowing navigation components to define an experimental condition.

This paper makes two contributions:

- RWD, an open-source robot-agnostic ROS 2 framework for deploying navigation systems through long-range, potentially multi-floor service robot tasks.
- An evaluation comprising a within-participant visitor-guidance comparison of different navigation systems and a cross-robot, cross-building deployment case study.

II. Related Work

Prior work relevant to RWD spans three areas: long-term indoor autonomy, navigation evaluation in simulation and the real world, and social navigation data collection.

A. Frameworks for Long-Term Indoor Robot Autonomy

Long-term autonomy systems establish that service robots can operate beyond isolated navigation trials. The STRANDS

project integrated navigation, recovery, spatiotemporal representations, and applications across deployments totaling 104 autonomous days and 116 km [12]. CoBots traveled more than 1,000 km while performing user-requested tasks in multi-floor buildings with human-assisted autonomy [13]. Francis et al. [11] noted that the robot system described by Xiao et al. [9] contributed to a Google data-collection deployment exceeding 3,000 km. Although STRANDS released substantial open-source software targeting end-of-life ROS 1 distributions and SCITOS robots, the other systems do not provide comparable public access. By contrast, RWD is designed as a hardware-agnostic, open-source framework for indoor service-robot tasks, using Nav2 [14] and BehaviorTree.CPP [15] for task execution.

Long-term deployment also depends on high-level task planning and dispatch. PlanSys2 provides general PDDL-based task planning and BT execution [16]. Door traversal and elevator use can be expressed in PlanSys2 but require a domain model and corresponding performers. Open-RMF [17] provides task allocation and traffic management for heterogeneous fleets. RWD complements these capabilities by focusing on task execution and indoor autonomy behaviors for individual robots, with additional components supporting social navigation research.

B. Navigation Evaluation in Simulation and the Real World

Classical and learned navigation methods divide perception, planning, and control differently. Xiao et al. [18] found that most surveyed learning-based methods were end-to-end, while direct comparisons with classical methods were uncommon and frequently limited to simple settings. RWD targets long-range missions with candidate methods implementing the Nav2 planner and controller interfaces.

Simulation complements real-world navigation evaluation by enabling repeatable trials before deployment. HuNavSim and Arena provide social navigation scenarios, human models, and metrics in ROS 2 simulation [3], [4]. Francis et al. [11] concluded that simulation supports issue discovery but does not replace real-world testing. Open-RMF [17] provides tools for building simulated multi-floor office buildings and hospitals. Drawing on these systems, RWD provides a Gazebo-based simulation workflow that replicates target deployment environments.

C. Social Navigation Data Collection

Human-generated datasets provide reference behaviors and diverse scenarios, but their collection is labor-intensive. SCAND [10] used teleoperated robots, MuSoHu [19] equipped walking participants with an egocentric sensor suite, and ACME [20] extended human-demonstrated data across cultures and robot platforms. THÖR-MAGNI [21] and Bi^3 [22] provided close-range human-robot interaction in deliberately bounded spaces and tightly controlled scenarios, with the latter evaluating four navigation controllers. HuRoN [23] collected more than 75 hours and 4,000 interactions autonomously, coupling data acquisition with a learned policy. Francis et al. [11] identified narrow doorways and elevator entry and exit as important social navigation scenarios. RWD supports data collection for these scenarios through explicit door and elevator behaviors.

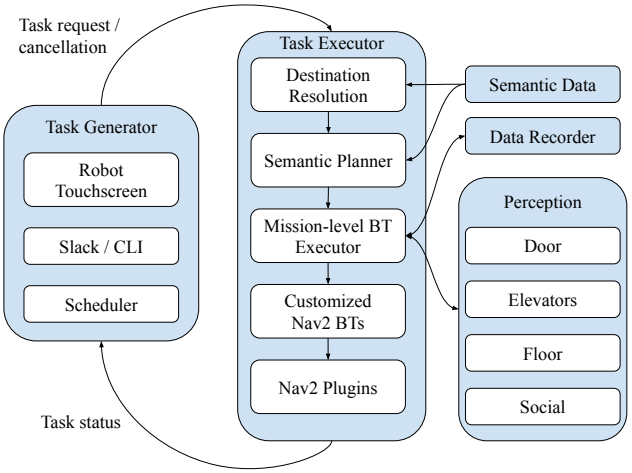


Fig. 2: RWD architecture separating reusable components from configurable building data, hardware, perception, and navigation.

III. RWD Framework

RWD provides an end-to-end workflow for indoor service-robot operation, from user-submitted tasks through semantic planning, robot motion, operational monitoring, and task-associated data collection. This section presents the framework's design boundary, core task execution components, supporting components, and deployment workflow.

A. Design Boundary and Architecture

RWD provides a reusable layer for planning and executing indoor service tasks. It standardizes semantic mission planning, task lifecycle, building interactions, operational monitoring, and data collection, while robot hardware, building data, perception algorithms, and navigation plugins remain replaceable (Fig. 2).

RWD represents package delivery, visitor guidance, and scheduled patrol tasks as waypoint-based navigation missions. The onboard touchscreen, two-way Slack interface, and programmatic task scheduler use a common interface to submit and cancel tasks. The mission planner combines the requested destination, current location, and building semantic data into a multi-step sequence. The BT-based task executor owns the task lifecycle and dispatches each step through a Nav2 action or the corresponding building-interaction behavior. Building semantics and route constraints remain independent of navigation configuration and can be reused across compatible robot platforms.

At minimum, a robot system provides a velocity-command interface, onboard compute for ROS 2 with Nav2, and the sensor inputs required by its targeted navigation systems. Task clients and navigation systems may require an onboard touchscreen, RGB cameras, or network connectivity. RWD

provides a template adapter package to match robot-specific hardware configurations.

The reference deployment with an AgileX Scout Mini robot (Fig. 1) uses human-assisted autonomy because the mobile base cannot manipulate doors or request elevators. The task executor presents assistance requests on the onboard touchscreen and in Slack, then proceeds when the corresponding perception provider reports that the transition is available. Robots with manipulation capabilities and buildings with API-enabled doors or elevators can retain the same semantic mission planning but need custom behavior subtrees for their interactions.

B. Building Semantics and Mission Planning

RWD provides tools for constructing coherent robot-agnostic building semantics from a 2D occupancy-grid map for each floor. The resulting representation associates each floor map with semantic waypoints for rooms, route constraints, and door or elevator transitions, then connects those waypoints in a sparse semantic graph $G = (V, E)$. An edge-type function $\tau : E \to \{\texttt{navigate}, \texttt{door}, \texttt{elevator}\}$ and a nonnegative cost function $w : E \to [0, \infty)$ describe graph transitions, while a map function $\mu : V \to \mathcal{M}$ associates each waypoint with an occupancy map in the building's map set $\mathcal{M}$. Door and elevator entities describe the corresponding building interactions, while rigid 2D floor alignments support localization transfer after elevator travel.

For each task, the mission planner applies Dijkstra's algorithm to find a minimum-cost route $P = (e_1, \ldots, e_n)$ from the robot's current semantic location to the requested destination (Fig. 3). It compiles the route's navigation, door, and elevator edges into an immutable sequence of mission steps.

C. Waypoint-Based Navigation and Task Constraints

RWD defines waypoints as map-based 2D poses that reference rooms, areas, support routing, or locations associated with building interactions. Two design goals determine how waypoint types constrain a mission.

First, `routing_only` waypoints shape graph search without becoming execution goals, while `must_pass` waypoints retain loose intermediate constraints for required corridors or to prevent unintended shortcuts. The CollapseNavigationRun procedure (Alg. 1, lines 3–8) retains these constraints in M and compiles each maximal same-map run of ordinary navigation edges into one mission step (line 9). The task executor uses `NavigateToPose` if M is empty and `NavigateThroughPoses` otherwise, allowing geometric freedom between required constraints (Fig. 3).

Second, semantic door and elevator waypoints place explicit boundaries around building-interaction behaviors. Door and elevator edges terminate a collapsible navigation sequence because they require dedicated interaction behavior. After a door opens, the opposite-side waypoint can remain a loose intermediate constraint so the robot moves smoothly toward its next stopping goal, or it can become an exact stopping goal before ordinary navigation resumes.

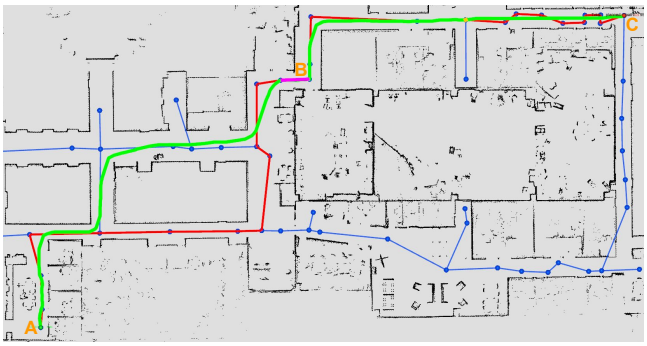


Fig. 3: Single-floor mission from A to C. Blue denotes the semantic graph, red the selected graph route, green the Theta* path [24], and magenta door traversal at B. Navigation-run collapse retains `must_pass` constraints while allowing geometric planning between them. The resulting path is 96 m long.

Algorithm 1 CollapseNavigationRun

Input: A maximal same-map run $R = (r_1, \ldots, r_k)$ of `navigate` edges.
Output: One navigation mission step S.

1 $s \leftarrow \text{source}(r_1)$, $g \leftarrow \text{target}(r_k)$, and $M \leftarrow \langle\,\rangle$.
2 $c \leftarrow \sum_{i=1}^{k} w(r_i)$.
3 **for** $i \leftarrow 1$ **to** $k - 1$ **do**
4 $v \leftarrow \text{target}(r_i)$.
5 **if** $\text{waypointType}(v) = \texttt{must_pass}$ **then**
6 append v to M.
7 **end if**
8 **end for**
9 **return** $S \leftarrow \text{NavigationStep}(s, g, M, c)$.

D. BT-Based Task Execution and Building Interaction

For each accepted task request, the BT-based task executor creates a task UUID, acquires an immutable multi-step mission plan, and advances the execution after each completed step. Its mission-level tree, implemented with BehaviorTree.CPP [15], uses memory sequences to preserve completed work and a reactive sequence to reevaluate task guards and recording state. A fallback dispatches the current navigation, door, or elevator step using the corresponding subtree.

For building interactions like door traversal and elevator ride, the branches share a common pattern. They first navigate to an approach pose, activate the configured perception provider, request assistance as needed, wait for a permissive observation, cross the transition autonomously, and release the provider. Activating providers only while their observations are required avoids continuous computation and power consumption. A common assistance interface exposes door and elevator requests to the Slack gateway and onboard GUI, allowing nearby people to assist without robotics expertise and supporting future development with automatic building features or mobile manipulation.

As actual physical environments may vary greatly, RWD standardizes perception provider start/stop services and observation messages so each deployment can use algorithms

suited to its sensors and environment. The reference suite includes lidar-based door state estimation, a surround-view RGB-based classifier for open elevator detection from building auxiliary features, and after-elevator floor detection using a vision-language model API. For lower autonomy level in challenging scenarios, an onboard GUI-based provider can accept operator input for these perception states and transitions.

E. Nav2 Plugins and Experimental Boundary

RWD uses Nav2's collection of global planners, local controllers, goal checkers, recovery behaviors, and action-level behavior trees through standard Nav2 actions [14]. Configuration selects these components for ordinary navigation or constrained door and elevator motion without changing the semantic mission or task lifecycle. RWD propagates Nav2's BT execution status back to the mission-level BT, allowing the task executor to advance or terminate the mission step without blocking.

Navigation algorithms developed outside Nav2 can be adapted to its standard planner interface, controller interface, or both. Methods that produce geometric paths fit the planner interface, methods that produce velocity commands fit the controller interface, and methods that jointly plan and control may expose both interfaces.

F. Data Recording and Feedback Collection

RWD supports selective recording under onboard storage constraints. Recent social navigation methods filter scenes without human presence [25], [26]. The mission-level BT starts and stops a ROS 2 bag recorder with configurable topics. Recording reasons include human presence, door traversal, and elevator transitions. Recording starts when at least one reason is active and stops when none remains. The recorded bags include task metadata and navigation settings.

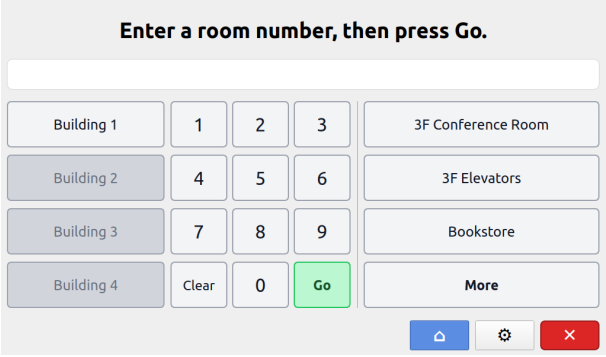
Aside from robot data, RWD also gathers post-task ratings and feedback through the onboard touchscreen (Fig. 4). Each response is associated with the task UUID, allowing a low rating to identify the corresponding task execution and sensor recordings for detailed review.

G. Operational Monitoring

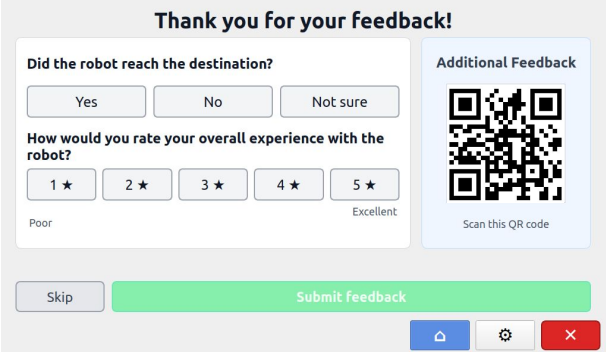
The common task and assistance interfaces provided by RWD enable remote supervisors to monitor progress, cancel or request tasks, and respond to robot requests. Optional deployment helpers integrate an existing compressed camera topic with off-the-shelf RTSP and WebRTC components for remote video monitoring over a local network or a separately managed private VPN. These tools do not participate in task execution or navigation.

H. Deployment Workflow

RWD uses a real-to-sim-to-real workflow. First, developers create 2D occupancy maps with a SLAM system such as SLAM Toolbox. RWD map tools add semantic task information (see Fig. 6), while other open-source tools can convert the 2D maps into Gazebo worlds.



(a) Task request.



(b) Feedback collection.

Fig. 4: Onboard touchscreen interfaces for task requests (a) and post-task feedback (b).

Next, coarse measurements align doors and elevators with their semantic waypoints and place the corresponding interactive assets in simulation (see Fig. 5), similar to the explicit site modeling used by Open-RMF [17]. The simulator reuses the semantic data and task interfaces intended for the physical deployment, so an ordinary destination request exercises the complete configured task path, including building interactions.

Simulation exposes failures in waypoint placement, routing, building interaction setup, floor alignment, and navigation through constrained spaces. Developers refine semantic configuration and navigation settings in simulation before final adjustments through real-robot testing.

IV. Experiments and Evaluation

The evaluation uses a visitor guidance study to compare three navigation systems and a deployment case study to measure transfer effort across robots and buildings.

A. Visitor Guidance User Study

Research question. Can RWD support a controlled, within-participant comparison of user experience across navigation systems during visitor guidance?

a) Navigation system conditions: The three conditions pair a common Theta* global planner [24] with the non-social DWB baseline [14], the rule-based Social Force Window planner (SFW) [27], and the learned DRL-VO social navigation controller [28]. The study initially considers the

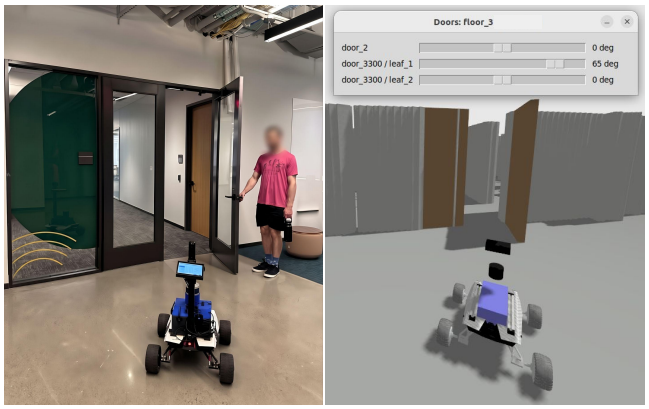


Fig. 5: **Left**: The robot waiting for human assistance at a door. **Right**: Same scenario in the corresponding Gazebo world with RWD providing manual door control.

visual foundation model ViNT [29], but excludes it after pilot testing because the adapted implementation, trained on targeted image-based topological route, still requires frequent safety interventions and is sensitive to lighting changes.

All three controllers run at 10 Hz with a shared $0.8\,\mathrm{m/s}$ maximum speed. A forward-facing ZED 2 camera and the ZED SDK provide tracked pedestrian positions and velocities to SFW and DRL-VO. Simulation-based tuning addresses static-environment challenges, primarily tight corners and motion smoothness. Other parameter values remain consistent with the reference implementations.

b) Study protocol: The study uses a $135\,\mathrm{m}$ indoor route across connected buildings, moderately crowded areas, blind corners, and narrow and wide corridors. The study includes 15 participants, each completing the route once per system in counterbalanced order. Participants request tasks through the onboard touchscreen and follow the robot (Fig. 1), and the operator switches controllers between trials. A trial ends when the robot reaches the destination or task execution reports failure. A remote switch allows the operator to override controller velocity commands for safety interventions. The study follows an IRB-approved protocol, with uncompensated volunteers in a campus-building setting.

Seven five-point agreement items adapted from Pirk et al. [30] assess safe distance (SD), ease of following (EF), intent clarity (IC), social appropriateness (SA), obstacle navigation (ON), completion time (CT), and overall rating (OR). Scores range from 1 (least favorable) to 5 (most favorable). Participants respond after each trial and before the next condition.

c) Questionnaire results and interpretation: Table I summarizes mean ratings across 45 trials. Exact Friedman tests with Holm correction do not reject the null hypotheses of no controller-related rating differences for any of the seven items at $\alpha = 0.05$.

Despite behavioral differences noted by participants with robotics backgrounds, the results leave open whether these local controllers meaningfully affect user experience during service-robot tasks. This small study motivates larger, longer-term evaluations.

TABLE I: Mean visitor guidance ratings (1–5).

Controller	SD	EF	IC	SA	ON	CT	OR
DWB	3.47	4.00	3.07	3.33	3.67	4.31	3.87
DRL-VO	3.47	3.73	3.13	3.40	3.73	4.15	3.67
SFW	3.80	3.47	3.07	3.33	3.40	3.54	3.33

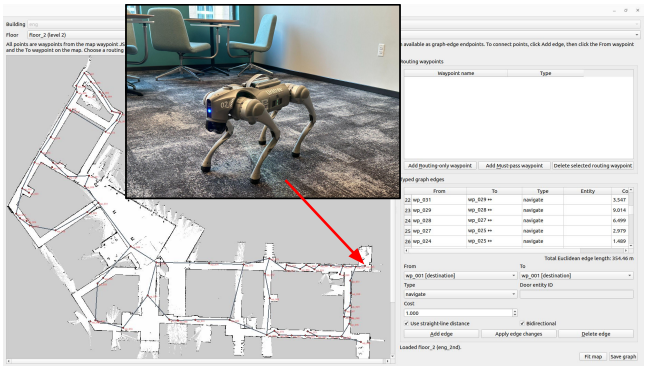


Fig. 6: Use of a RWD semantic configuration tool on a new floor map in the case study with Unitree Go2 robot.

d) Framework operation: RWD components operate without technical issues across all 45 trials. Trials average roughly 3 minutes, and the three trials and surveys take approximately 15 minutes per participant.

B. Cross-Robot and Cross-Building Deployment Case Study

Research question. How much active engineering time is required to deploy RWD on a Unitree Go2 (Fig. 6) and configure building semantics in a different building? Starting from a clean checkout, the case study proceeds through integration and configuration to cross-floor tasks involving doors and an elevator. A task succeeds when the robot reaches the requested destination through the standard task interface, mission planner, and BT-based task executor without a safety intervention.

A work log assigns active person-hours to the five stages in Table II. Active time includes attended mapping, annotation, implementation, tuning, and diagnosis but excludes unattended builds, downloads, and charging. The real-world validation stage includes one navigation round through all configured door and elevator transitions and additional tasks covering most configured paths on each floor.

The RWD template adapter integrates the Go2 through an open-source ROS 2 wrapper. Topic and robot-parameter configuration requires one active person-hour, after which the robot executes tasks submitted from a tethered laptop in the primary testing environment.

The cross-building evaluation covers public routes in a five-story, approximately $16{,}700\,\mathrm{m}^2$ academic building. Its semantic configuration includes 323 classrooms, offices, labs, and common areas, 26 doors, one elevator lobby with 3 cars, and approximately $1{,}350\,\mathrm{m}$ of route coverage (Fig. 6).

After real-world validation, the evaluation uses 10 start–destination pairs with average path distance of 55 meters

TABLE II: Configuration effort to deploy RWD in a new five-story building.

Stage	Active person-hours
Robot integration	1.0
Map creation	1.5
Semantic data creation	3.0
Tuning in simulation	1.5
Real-world validation	2.0
Total	9.0

for testing. Nine of the 10 tasks (90%) reach their destinations without safety intervention while using the configured human-assisted door and elevator behaviors. The remaining task is stopped to avoid a potential collision with a glass wall.

V. Discussion

This section discusses the research uses of RWD, its current limitations, and directions for broader deployment.

A. Interpretation and Potential Applications

RWD treats a complete long-range service mission as the unit of navigation evaluation. The shared task UUID connects each outcome with its semantic context, interventions, user feedback, and sensor recordings.

The common Nav2 boundary keeps the surrounding task infrastructure fixed while preserving each navigation system as a complete experimental condition. Any observed difference therefore reflects the complete navigation system condition, including its planner or controller, adapter, inputs, parameters, and tuning.

Explicit building-interaction boundaries extend navigation evaluation to social behavior at doors and elevators. Door behaviors can incorporate compatible doorway-traversal or manipulation methods within a complete service task [31]. Elevator behaviors provide a common execution context for studying waiting, boarding, yielding, and crowd-dependent acceptance [32]. Developing these behaviors identifies three perception problems that need more general solutions: detecting whether a door is open, identifying which elevator has arrived, and confirming a floor change.

Longer deployments also turn routine service tasks into structured social navigation data collection. These records can support studies of pedestrian flow, local passing conventions, and online navigation adaptation, including methods such as OLiVia-Nav and SOCSARL-OL [33], [34]. After continued deployments provide sufficient coverage across routes, navigation systems, and building interactions, the project will release the resulting dataset with its task-associated metadata.

RWD can further support robots with richer social expression, such as Quori [35]. An RWD adapter for this platform can combine navigation with speech, gaze, or gesture while retaining the task execution and data collection interfaces.

B. Limitations

RWD currently has three main framework limitations. First, its semantic representation assumes that every waypoint and building interaction can be expressed relative to a stable 2D map pose. Consequently, RWD does not yet directly support robots that rely on 3D, visual, or purely topological maps and localization.

Second, deployment still requires site-specific configuration and robot integration. Developers must create or import maps, annotate building semantics, configure perception providers, connect the robot adapter, and tune navigation for the platform and environment.

Third, the current Gazebo workflow primarily supports system integration rather than high-fidelity navigation evaluation. Generated worlds reproduce the occupancy geometry and selected building interactions, but they do not yet reproduce realistic pedestrian behavior, detailed visual appearance, or all physical sensor effects.

C. Future Work

Future work will introduce map and localization backends that separate semantic task entities from the current 2D pose representation. These backends will allow the same mission planner and task executor to reference richer 3D, visual, or topological representations while preserving explicit destinations and building-interaction boundaries.

The simulation workflow will also incorporate configurable pedestrian behavior and photorealistic environment and sensor rendering, building on social-navigation simulators such as HuNavSim and Arena [3], [4]. Reusing the same semantic missions across these simulators and the physical robot will support controlled social and visual navigation studies before deployment.

Long-term execution records can provide time-dependent estimates of edge availability and traversal cost for corridors, doors, and elevators. Future work will incorporate these estimates into semantic route selection so the planner can prefer routes that are more likely to be available and reliable under the current operating conditions.

Future work will also extend assistance events into inter-robot requests. A mobile manipulator can operate a door or elevator for a mobile base, while an Open-RMF adapter can coordinate task assignment and shared building resources across a fleet [17].

VI. Conclusion

RWD provides reusable software components for deploying and evaluating social navigation through long-range indoor service tasks. The framework combines building semantics, waypoint-based mission planning, BT-based task execution, building-interaction interfaces, operational monitoring, and task-associated data collection. The evaluation compares three navigation systems during visitor guidance and records the effort required to configure the framework for a Unitree Go2 and deploy in a large academic building.

REFERENCES

[1] E. Ackerman, "How diligent's robots are making a difference in texas hospitals," *IEEE Spectrum*, March 2020. [Online]. Available: https://spectrum.ieee.org/how-diligents-robots-are-making-a-difference-in-texas-hospitals

[2] Serve Robotics, "Serve robotics: Autonomous sidewalk delivery robots," https://www.serverobotics.com/, accessed: 2026-08-28.

[3] N. Pérez-Higueras, R. Otero, F. Caballero, and L. Merino, "Hunavsim: A ros 2 human navigation simulator for benchmarking human-aware robot navigation," *IEEE robotics and automation letters*, vol. 8, no. 11, pp. 7130–7137, 2023.

[4] V. Shcherbyna, L. Kastner, D. Diaz, H. G. Nguyen, M. H.-K. Schreff, T. Seeger, J. Kreutz, A. Martban, Z. Shen, H. Zeng, *et al.*, "Arena 4.0: A comprehensive ros2 development and benchmarking platform for human-centric navigation using generative-model-based environment generation," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 9138–9144.

[5] A. Stratton, P. T. Singamaneni, P. Goyal, R. Alami, and C. Mavrogiannis, "How human motion prediction quality shapes social robot navigation performance in constrained spaces," in *Proceedings of the 21st ACM/IEEE International Conference on Human-Robot Interaction*, 2026, pp. 1060–1069.

[6] D. Song, J. Liang, A. Payandeh, A. H. Raj, X. Xiao, and D. Manocha, "Vlm-social-nav: Socially aware robot navigation through scoring using vision-language models," *IEEE Robotics and Automation Letters*, vol. 10, no. 1, pp. 508–515, 2024.

[7] A. Payandeh, D. Song, M. Nazeri, J. Liang, P. Mukherjee, A. H. Raj, Y. Kong, D. Manocha, and X. Xiao, "Social-llava: Enhancing social robot navigation through human-language reasoning," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025, pp. 17 192–17 198.

[8] A. H. Raj, Z. Hu, H. Karnan, R. Chandra, A. Payandeh, L. Mao, P. Stone, J. Biswas, and X. Xiao, "Rethinking social robot navigation: Leveraging the best of two worlds," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 16 330–16 337.

[9] X. Xiao, T. Zhang, K. M. Choromanski, T.-W. E. Lee, A. Francis, J. Varley, S. Tu, S. Singh, P. Xu, F. Xia, S. M. Persson, D. Kalashnikov, L. Takayama, R. Frostig, J. Tan, C. Parada, and V. Sindhwani, "Learning model predictive controllers with real-time attention for real-world navigation," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 1708–1721. [Online]. Available: https://proceedings.mlr.press/v205/xiao23a.html

[10] H. Karnan, A. Nair, X. Xiao, G. Warnell, S. Pirk, A. Toshev, J. Hart, J. Biswas, and P. Stone, "Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 807–11 814, 2022.

[11] A. Francis, C. Pérez-d'Arpino, C. Li, F. Xia, A. Alahi, R. Alami, A. Bera, A. Biswas, J. Biswas, R. Chandra, *et al.*, "Principles and guidelines for evaluating social robot navigation algorithms," *ACM Transactions on Human-Robot Interaction*, vol. 14, no. 2, pp. 1–65, 2025.

[12] N. Hawes, C. Burbridge, F. Jovan, L. Kunze, B. Lacerda, L. Mudrova, J. Young, J. Wyatt, D. Hebesberger, T. Kortner, *et al.*, "The strands project: Long-term autonomy in everyday environments," *IEEE Robotics & Automation Magazine*, vol. 24, no. 3, pp. 146–156, 2017.

[13] M. M. Veloso, J. Biswas, B. Coltin, and S. Rosenthal, "CoBots: Robust symbiotic autonomous mobile service robots," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015, pp. 4423–4429. [Online]. Available: https://www.ijcai.org/Proceedings/15/Papers/656.pdf

[14] S. Macenski, F. Martín, R. White, and J. G. Clavero, "The marathon 2: A navigation system," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 2718–2725.

[15] D. Faconti, "Behaviortree.cpp," https://github.com/BehaviorTree/BehaviorTree.CPP, 2018–2026. [Online]. Available: https://www.behaviortree.dev/

[16] F. Martín, J. G. Clavero, V. Matellán, and F. J. Rodríguez, "Plansys2: A planning system framework for ros2," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 9742–9749.

[17] Open Source Robotics Foundation, "Open-RMF: A common language for robot interoperability," Open-RMF project website, accessed: 2026-08-22. [Online]. Available: https://www.open-rmf.org/

[18] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion planning and control for mobile robot navigation using machine learning: a survey," *Autonomous Robots*, vol. 46, no. 5, pp. 569–597, 2022.

[19] D. M. Nguyen, M. Nazeri, A. Payandeh, A. Datar, and X. Xiao, "Toward human-like social robot navigation: A large-scale, multi-modal, social human navigation dataset," in *2023 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2023, pp. 7442–7447.

[20] S. R. Marpally, A. Wang, A. Ghotavadekar, R. A. Ribeiro, N. Le, P. Bachiller-Burgos, P. Goyal, S. Agrawal, Y. Nitta, H. Z. Han, *et al.*, "Acme: A multi-cultural, multi-embodiment social-navigation dataset," *arXiv preprint arXiv:2607.21964*, 2026.

[21] T. Schreiter, T. Rodrigues de Almeida, Y. Zhu, E. Gutierrez Maestro, L. Morillo-Mendez, A. Rudenko, L. Palmieri, T. P. Kucner, M. Magnusson, and A. J. Lilienthal, "Thör-magni: a large-scale indoor motion capture recording of human movement and robot interaction," *The International Journal of Robotics Research*, vol. 44, no. 4, pp. 568–591, 2025.

[22] A. Stratton, P. T. Singamaneni, P. Goyal, R. Alami, and C. Mavrogiannis, "Bi^3: A biplatform, bicultural, biperson dataset for social robot navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2026. [Online]. Available: https://arxiv.org/abs/2605.06863

[23] N. Hirose, D. Shah, A. Sridhar, and S. Levine, "SACSoN: Scalable autonomous control for social navigation," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 49–56, 2024.

[24] K. Daniel, A. Nash, S. Koenig, and A. Felner, "Theta*: Any-angle path planning on grids," *Journal of Artificial Intelligence Research*, vol. 39, pp. 533–579, 2010.

[25] D. Song, N. Le, J. Chen, M. Nazeri, A. Payandeh, R. Chandra, R. Mirsky, R. Mead, L. Xiao, and X. Xiao, "Humain: Human-aware implicit social robot navigation," 2026. [Online]. Available: https://arxiv.org/abs/2607.07357

[26] A. Payandeh, A. Pokhrel, D. Song, M. Zampieri, and X. Xiao, "Narrate2nav: Real-time visual navigation with implicit language reasoning in human-centric environments," 2025. [Online]. Available: https://arxiv.org/abs/2506.14233

[27] N. Pérez-Higueras and R. Otero, "Social force window planner," GitHub repository, 2023, version 0.4.7; accessed 2026-09-13. [Online]. Available: https://github.com/robotics-upo/social_force_window_planner

[28] Z. Xie and P. Dames, "DRL-VO: Learning to navigate through crowded dynamic scenes using velocity obstacles," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2700–2719, 2023.

[29] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine, "ViNT: A foundation model for visual navigation," in *Proceedings of the 7th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 229. PMLR, 2023, pp. 711–733.

[30] S. Pirk, E. Lee, X. Xiao, L. Takayama, A. Francis, and A. Toshev, "A protocol for validating social navigation policies," *arXiv preprint arXiv:2204.05443*, 2022.

[31] A. Snowdy, A. Trivedi, S. Prajapati, L. M. Genua, and T. Padir, "Socially-aware autonomous doorway traversal and payload delivery for emergency assistance," 2026. [Online]. Available: https://arxiv.org/abs/2607.05315

[32] S. Kim, S. Cho, K. Kim, and S. Bak, ""too crowded for a robot?": Modeling human acceptance criteria for elevator-riding robots," in *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*, 2026, pp. 1–15.

[33] S. Narasimhan, A. H. Tan, D. Choi, and G. Nejat, "OLiVia-Nav: An online lifelong vision language approach for mobile robot social navigation," in *2025 IEEE International Conference on Robotics and Automation*, 2025, pp. 9130–9137.

[34] I. Okunevich, A. Lombard, T. Krajník, Y. Ruichek, and Z. Yan, "Online context learning for socially compliant navigation," *IEEE Robotics and Automation Letters*, vol. 10, no. 5, pp. 5042–5049, 2025.

[35] A. Specian, R. Mead, S. Kim, M. Matarić, and M. Yim, "Quori: A community-informed design of a socially interactive humanoid robot," *IEEE Transactions on Robotics*, vol. 38, no. 3, pp. 1755–1772, 2021.