Autonomous Ground Navigation in Highly Constrained Spaces: Lessons learned from The BARN Challenge at ICRA 2022

Competition Organizers: Xuesu Xiao[1,2,3], Zifan Xu[3], Zizhao Wang[3], Yunlong Song[4], Garrett Warnell[3,5], Peter Stone[3,6], Tingnan Zhang[7],
Finals Participants: Shravan Ravi[3], Gary Wang[3], Haresh Karnan[3], Joydeep Biswas[3], Nicholas Mohammad[8], Lauren Bramblett[8], Rahul Peddi[8], Nicola Bezzo[8], Zhanteng Xie[9], and Philip Dames[9]

***Abstract*— The BARN (Benchmark Autonomous Robot Navigation) Challenge took place at the 2022 IEEE International Conference on Robotics and Automation (ICRA 2022) in Philadelphia, PA. The aim of the challenge was to evaluate state-of-the-art autonomous ground navigation systems for moving robots through highly constrained environments in a safe and efficient manner. Specifically, the task was to navigate a standardized, differential-drive ground robot from a predefined start location to a goal location as quickly as possible without colliding with any obstacles, both in simulation and in the real world. Five teams from all over the world participated in the qualifying simulation competition, three of which were invited to compete with each other at a set of physical obstacle courses at the conference center in Philadelphia. The competition results suggest that autonomous ground navigation in highly constrained spaces, despite seeming ostensibly simple even for experienced roboticists, is actually far from being a solved problem. In this article, we discuss the challenge, the approaches used by the top three winning teams, and lessons learned to direct future research.**

I. The BARN Challenge Overview

Designing autonomous robot navigation systems has been a topic of interest to the robotics community for decades [1]–[5]. Indeed, there currently exist many such systems that allow robots to move from one point to another in a collision-free manner (e.g., open-source implementations in the Robot Operating System (ROS) [4]–[6] with extensions to different vehicle types [7]), which may create the perception that autonomous ground navigation is a solved problem. This perception may be reinforced by the fact that many mobile robot researchers have moved on to orthogonal navigation problems [8] beyond the traditional metric (geometric) formulation that only focuses on path optimality and obstacle avoidance. These orthogonal problems include, among others, learning navigation systems in a data-driven manner [9]–[12], navigating in off-road [13]–[15] and social contexts [16]–[18], and multi-robot navigation [19], [20].

However, autonomous mobile robots still struggle in many *ostensibly* simple scenarios, especially during real-world deployment [21]–[24]. For example, even when the problem is simply formulated as traditional metric navigation so that the only requirement is to avoid obstacles on the way to the goal, robots still often get stuck or collide with obstacles when trying to navigate in naturally cluttered daily households [21], in constrained outdoor structures including narrow walkways and ramps [23], [24], and in congested social spaces like classrooms, offices, and cafeterias [22]. In such scenarios, extensive engineering effort is typically required in order to deploy existing approaches, and this requirement presents a challenge for large-scale, unsupervised, real-world robot deployment. Overcoming this challenge requires systems that can both successfully and efficiently navigate a wide variety of environments with confidence.

The Benchmark Autonomous Robot Navigation (BARN) Challenge [25] was a competition at the 2022 IEEE International Conference on Robotics and Automation (ICRA 2022) in Philadelphia, PA that aimed to evaluate the capability of state-of-the-art navigation systems to solve the above-mentioned challenge, especially in highly-constrained environments, where robots need to squeeze between obstacles to navigate to the goal. To compete in The BARN Challenge, each participating team needed to develop an entire software stack for navigation for a standardized and provided mobile robot. In particular, the competition provided a Clearpath Jackal [26] with a 2D 270°-field-of-view Hokuyo LiDAR for perception and a differential drive system with 2m/s maximum speed for actuation. The aim of each team was to develop navigation software stack needed to autonomously drive the robot from a given starting location through a dense obstacle filed and to a given goal, and to accomplish this task without any collisions with obstacles or any human interventions. The team whose system could best accomplish this task within the least amount of time would win the competition. The BARN Challenge had two phases: a qualifying phase evaluated in simulation, and a final phase evaluated in a set of physical obstacle courses. The qualifying phase took place before the ICRA 2022 conference using the BARN dataset [27], which is composed of 300 obstacle courses in Gazebo simulation randomly generated by cellular automata. The top three teams from the simulation phase were then invited to compete in three different physical obstacle courses set up by the organizers at ICRA 2022 in the Philadelphia Convention Center.

In this article, we report on the simulation qualifier and physical finals of The BARN Challenge at ICRA 2022,

[1]George Mason University [2]Everyday Robots [3]The University of Texas at Austin [4]University of Zurich [5]Army Research Laboratory [6]Sony AI [7]Robotics@Google [8]University of Virginia [9]Temple University

present the approaches used by the top three teams, and discuss lessons learned from the challenge that point out future research directions to solve the problem of autonomous ground navigation in highly constrained spaces.

II. Simulation Qualifier

The BARN Challenge started on March 29th, 2022, two months before the ICRA 2022 conference, with a standardized simulation qualifier. The qualifier used the BARN dataset [27], which consists of 300 $5\text{m} \times 5\text{m}$ obstacle environments randomly generated by cellular automata (see examples in Fig. 1), each with a predefined start and goal. These obstacle environments range from relatively open spaces, where the robot barely needs to turn, to highly dense fields, where the robot needs to squeeze between obstacles with minimal clearance. The BARN environments are open to the public, and were intended to be used by the participating teams to develop their navigation stack. Another 50 unseen environments, which are not available to the public, were generated to evaluate the teams' systems. A random BARN environment generator was also provided to teams so that they could generate their own unseen test environments.[1]

In addition to the 300 BARN environments, six baseline approaches were also provided for the participants' reference, ranging from classical sampling-based [5] and optimization-based navigation systems [4], to end-to-end machine learning methods [28], [29], and hybrid approaches [30]. All baselines were implementations of different local planners used in conjunction with Dijkstra's search as the global planner in the ROS `move_base` navigation stack [31]. To facilitate participation, a training pipeline capable of running the standardized Jackal robot in the Gazebo simulator with ROS Melodic (in Ubuntu 18.04), with the option of being containerized in Docker or Singularity containers for fast and standardized setup and evaluation, was also provided.[2]

A. Rules

Each participating team was required to submit their developed navigation system as a (collection of) launchable ROS node(s). The challenge utilized a standardized evaluation pipeline[3] to run each team's navigation system and compute a standardized performance metric that considered navigation success rate (collision or not reaching the goal count as failure), actual traversal time, and environment difficulty (measured by optimal traversal time). Specially, the score s for navigating each environment i was computed as

$$s_i = 1_i^{\text{success}} \times \frac{\text{OT}_i}{\text{clip}(\text{AT}_i, 4\text{OT}_i, 8\text{OT}_i)} \ ,$$

where the indicator function 1_{success} evaluates to 1 if the robot reaches the navigation goal without any collisions, and evaluates to 0 otherwise. AT denotes the actual traversal time, while OT denotes the optimal traversal time, as an indicator of the environment difficulty and measured by the shortest traversal time assuming the robot always travels at its maximum speed (2m/s):

$$\text{OT}_i = \frac{\text{Path Length}_i}{\text{Maximal Speed}}.$$

The Path Length is provided by the BARN dataset based on Dijkstra's search from the given start to goal. The clip function clips AT within 4OT and 8OT, in order to assure navigating extremely quickly or slowly in easy or difficult environments respectively won't disproportionally scale the score. The overall score of each team is the score averaged over all 50 unseen test BARN environments, with 10 trials in each environment. Higher scores indicate better navigation performance. The six baselines score between 0.1627 and 0.2334. The maximum possible score based on our metric is 0.25.

B. Results

The simulation qualifier started on March 29th, 2022 and lasted through May 22th, 2022. In total, five teams from all over the world submitted their navigation systems. The performance of each submission was evaluated by a standard evaluation pipeline, and the results are shown in Tab. I.

TABLE I: Simulation Results

Rank.	Team/Method (University)	Score
1	TRAIL (Temple University)	0.2415
2	LfLH (Baseline [29])	0.2334
3	AMRL (UT Austin)	0.2310
4	AMR (UVA)	0.2200
5	E-Band (Baseline [4])	0.2053
6	End-to-End (Baseline [9])	0.2042
7	APPLR-DWA (Baseline [30])	0.1979
8	Yiyuiii (Nanjing University)	0.1969
9	NavBot (Indian Institute of Science)	0.1733
10	Fast (2.0m/s) DWA (Baseline [5])	0.1709
11	Default (0.5m/s) DWA (Baseline [5])	0.1627

All methods outperformed the baseline Dynamic Window Approach (DWA) [5], with both 2.0m/s and 0.5m/s max speed, the latter of which is the default local planner for the Jackal robot. However, only one approach (from Temple University) outperformed all baselines. The top three teams from the simulation qualifier, i.e., Temple Robotics and Artificial Intelligence Lab (TRAIL) from Temple University, Autonomous Mobile Robotics Laboratory from The University of Texas at Austin (AMRL UT Austin), and Autonomous Mobile Robots Lab from The University of Virginia (AMR UVA), were invited to the physical finals at ICRA 2022.

III. Physical Finals

The physical finals took place at ICRA 2022 in the Philadelphia Convention Center on May 25th and May 26th, 2022. Two physical Jackal robots with the same sensors and actuators were provided by the competition sponsor, Clearpath Robotics.

[1] https://github.com/dperille/jackal-map-creation
[2] https://github.com/Daffan/ros_jackal
[3] https://github.com/Daffan/nav-competition-icra2022

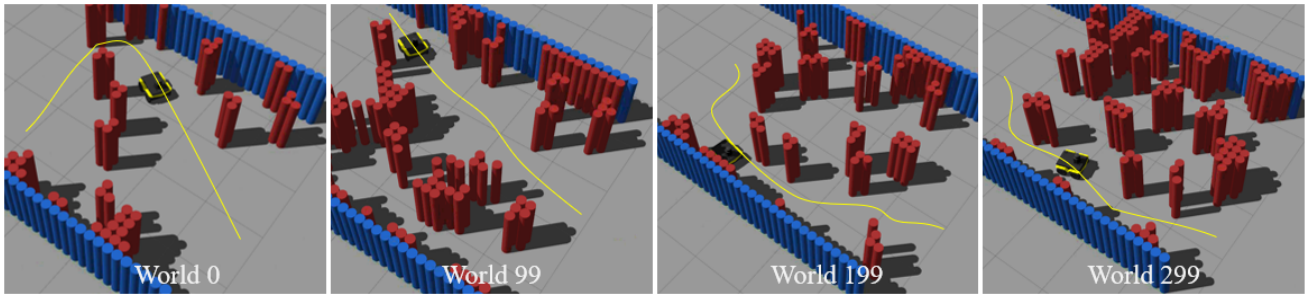


Fig. 1: Four Example BARN Environments in the Gazebo Simulator (ordered by ascending relative difficulty level)

.

Fig. 2: One (Out of Three) Physical Obstacle Courses during the Finals

A. Rules

Physical obstacle courses were set up using approximately 200 cardboard boxes in the convention center (Fig. 2). Because the goal of the challenge was to test a navigation system's ability to perform local planning, all three physical obstacle courses have an obvious passage that connects the start and goal locations (i.e., the robot shouldn't be confused by global planing at all), but the overall obstacle clearance when traversing this passage was designed to be very constrained, e.g., a few centimeters around the robot.

While it was the organizers' original intention to run exactly the same navigation systems submitted by the three top teams and use the same scoring metric in the simulation qualifiers in the physical finals, these systems suffered from (surprisingly) poor navigation performance in the real world (not even being able to finish one single trial without any collisions). Therefore, the organizers decided to change the rules by giving each team 30 minutes before competing in each of the three physical obstacle courses in order to fine-tune their navigation systems. After all three teams had this chance to set up for a particular obstacle course, the actual physical finals started as a 30-minute timed session for each team. In each 30-minute session, a team tested their navigation system in the obstacle course and notified the organizers when they were ready to time a competition trial. Each team had the opportunity to run five timed trials (after notifying the organizers). The fastest three out of the five timed trials were counted, and the team that had the most successful trials (reaching the goal without any collision) was the winner. In the case of a tie, the team with the fasted average traversal time would be declared the winner.

B. Results

The physical finals took place on May 25th and May 26th, 2022 (see the final award ceremony in Fig. 3). The three teams' navigation performance is shown in Tab. II. Since all navigation systems navigated at roughly the same speed, the final results were determined solely by the success rate of the best three out of five timed trials for each team. Surprisingly, the best system in simulation by Temple University exhibited the lowest success rate, while UT Austin's system enjoyed the highest rate of success.

Fig. 3: From Left to Right: Competition Sponsor (Clearpath Robotics), Competition Organizers, the Temple, UVA, and UT Austin teams

TABLE II: Physical Results

Rank.	Team/Method (University)	Success / Total Trials
1	AMRL (UT Austin)	8/9
2	AMR (UVA)	4/9
3	TRAIL (Temple University)	2/9

IV. Top Three Teams and Approaches

In this section, we report the approaches used by the three winning teams.

A. The University of Texas at Austin

To enable robust, repeatable, and safe navigation in constrained spaces frequently found in BARN, the UT

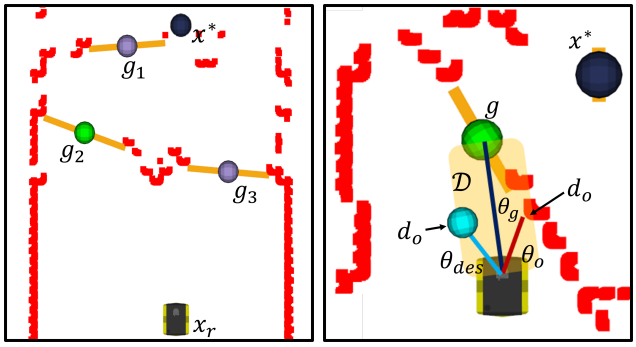
Austin team from AMRL[4] utilized state-of-the-art classical approaches to handle localization, planning, and control along with an automated pipeline to visualize and debug continuous integration. To plan feasible paths to reach the goal location while avoiding obstacles, a medium-horizon kinematic planner from `ROS move_base` [31] was used, combined with a discrete path rollout greedy planner for local kinodynamic planning from AMRL's graph navigation stack [32]. This two-stage hierarchical planning generated safe motion plans for the robot to make progress towards the goal while reactively avoiding obstacles along its path using the LiDAR scans. Additionally, since the environment contains tight spaces that are challenging to navigate through, it was observed that accurate motion estimation of the robot was crucial to deploying a planning-based navigation controller in an unmapped environment. When executing sharp turns in constrained environments, poor estimates of the robot's motion negatively interfered with costmap updates in `move_base` and often prevented the mid-level planner from discovering any feasible path to the goal.

Towards addressing this problem, Episodic non-Markov Localization (EnML) [33] was utilized, which fuses the LiDAR range scans with wheel odometry through non-markov bayesian updates. Combining EnML with two-stage hierarchical planning proved to be useful in safely handling constrained spaces. Additionally, the UT Austin team developed custom automated tools to generate visualizations for debugging that helped identify failure cases easily, perform manual hyperparameter tuning and accelerate bug fixes during the competition.

While classical approaches helped solve a majority of environments in the BARN challenge, significant challenges still remain for navigation in extremely constrained spaces. For example, the two-stage hierarchical planning module does not explore unobserved regions of the environment before committing to a kinematically feasible path. This sometimes leads to suboptimal paths causing longer time taken to reach the goal. We posit that a learnable mid-level planner with the ability to actively explore the environment appropriately to plan the optimal path may be a promising future direction of research to improve autonomous navigation in constrained spaces.

B. University of Virginia

In order to quickly and robustly navigate through the unknown, cluttered BARN challenge environments, the UVA AMR team[5] developed a mapless, "follow-the-gap" planning scheme which (a) detects open gaps for the robot to follow to reach a final goal and (b) plans local goals in order to reach these open gaps without colliding with intermediate obstacles. The framework expands upon the UVA AMR lab's previous work [34]. Fig. 4(a) illustrates the framework displaying the laser scan point-cloud of a world from the BARN dataset along with the detected intermediate gaps g_1, g_2, and g_3, vehicle position $x_r \in \mathbb{R}^2$, and final goal position $x^* \in \mathbb{R}^2$. Fig. 4(b) shows the local planner, which provides course corrections in order for the robot to avoid obstacles while reaching a selected gap goal. The



(a) (b)

Fig. 4: (UVA Team) Examples of (a) Detected Gaps in a Simulated BARN Environment and (b) Local Planner Obstacle Avoidance

approach takes advantage of the fact that gaps start or end at discontinuities in the laser scan and leverages this principle to find intermediate gap goals for navigation [35]. Let $p_i, p_{i+1} \in \mathbb{R}^2$ be adjacent points in the laser scan and R denote the maximum sensing range of the LiDAR. The first discontinuity, referred to as *Type 1*, occurs when the distance between the adjacent readings is larger than the circumscribed diameter d_r of the robot: $||p_i - p_{i+1}||_2 > d_r$. The second discontinuity, *Type 2*, occurs when one of the two readings is outside the LiDAR's sensing range: $||p_i - x_r||_2 \geq R \oplus ||p_{i+1} - x_r||_2 \geq R$. If $||p_{i+1} - x_r||_2 > ||p_i - x_r||_2$, the discontinuity is referred to as *rising*, otherwise it is *falling*. Below we describe how to leverage these discontinuities to identify gaps.

The first step in gap detection is to perform a forward pass from p_0 to p_{n-1} in the laser point-cloud scan for rising type 1 and type 2 discontinuities. Let p_i denote the location of the first rising discontinuity and $\mathcal{L}^+ = \{i+1, \ldots, n-1\}$. This point becomes the beginning of the gap. To determine the end, we find the point p_j closest to p_i such that $j \in \mathcal{L}^+$. That is,

$$p_j = \arg\min_{j \in \mathcal{L}^+} ||p_i - p_j||_2 \tag{1}$$

The process continues starting from p_{j+1}. Once the forward pass is complete, a mirrored backward pass from p_{n-1} to p_0 is done to find gaps via falling discontinuities. Each detected gap, defined as $g_i = (a_i, b_i)$, a tuple of the start and end points, are added to a quadtree $\mathcal{T}_g$ which keeps track of where all previously identified and visited gaps are located. If any gap already exists in the tree, it is ignored.

Once $\mathcal{T}_g$ is updated, a gap $g^* \in \mathcal{T}_g$ is selected to be the intermediate goal if it is determined that the final goal x^* is not *admissible*. In this context, admissibility is determined by checking if a given goal is navigable; that is, from the laser scan data, a path is known to exist from the robot position to

[4] `https://amrl.cs.utexas.edu/`
[5] `https://www.bezzorobotics.com/`

the goal. The check is done by using a similar algorithm as discussed by Minguez and Montano [36], which, given any start point x_a and end point x_b, ensures no inflated obstacles block the robot along the line connecting the two points.

The process to select the gap goal from $\mathcal{T}_g$ when x^* is inadmissible is outlined in Algorithm 1. At each iteration, the algorithm finds the closest gap g^* to the final goal x^*. If g^* is inadmissible from the robot's current position, properties of quadtree queries are utilized to find all gaps $G' \subseteq \mathcal{T}_g$ which must be passed as the robot drives from x_r to g^*. The algorithm then iteratively finds the closest admissible gap $g \in G'$ to the robot which is also admissible to g^*. Meaning, the robot knows that a feasible path from x_r to g and from g to g^* exists. If no g satisfy this constraint for the given g^*, the process repeats with g^* as the next closest gap to x^* and terminates once an admissible gap is found. For clarity, Fig. 4(a) shows an example of the goal selection process. The final goal x^* is not admissible, nor is the closest gap to it, g_1. However, x_r to g_2 is admissible as well as g_2 to g_1. Thus, g_2 is selected as the intermediate goal and the selection process repeats once the robot reaches g_2.

Algorithm 1 (UVA Team) Find Gap Goal

Input: quadtree $\mathcal{T}_g$, robot position x_r, final goal x^*
Output: gap goal g^*
while $\mathcal{T}_g \neq \emptyset$ & !isAdmissible(x_r, g^*) **do**
  $g^* \leftarrow \arg\min_{g^* \in \mathcal{T}_g} ||x^* - g^*||_2$
  $\mathcal{T}_g \leftarrow \mathcal{T}_g \setminus \{g^*\}$
  # Returns children in descending order of dist. to x_r
  $G' \leftarrow$ getChildren$(g^*, x_r, \mathcal{T}_g)$
  for $g \in G'$ **do**
    if isAdmissible(x_r, g) & isAdmissible(g, g^*) **then**
      $g^* = g$
    end if
  end for
end while
return g^*

Even though the selected gap goal is admissible, a direct path to it may not be feasible given the configuration of the obstacles within an environment. For example, a robot navigating directly to g in Fig. 4(b) will collide with the obstacles shown by the laser scan data. In order to prevent such issues from arising, local planner is utilized which re-plans the mobile robot's trajectory at every timestep if collision is imminent. The direct path to the goal is formulated as a region $\mathcal{D}$, which accounts for the relative heading to the goal, θ_g and the diameter d_r of the robot. The region $\mathcal{D}$ is checked against the laser scan points for any obstacles; let $\boldsymbol{p}$ represent all obstacle coordinates within region $\mathcal{D}$. If no obstacles are in $\mathcal{D}$, that is $\boldsymbol{p} = \emptyset$, the robot is sent directly to the gap goal, g. If there are multiple obstacles within $\mathcal{D}$, the one closest to the robot is selected; let d_o represent the distance to the closest obstacle and θ_o represent the direction of the obstacle with respect to the robot's heading. The new desired heading is then computed by accounting the offset between goal and obstacle to the gap goal: $\theta_{des} = \theta_g + (\theta_g - \theta_o)$, and the local goal is placed at a distance of d_o in this desired direction (shown in teal in Fig. 4(b)).

The inputs to the robot are angular and linear velocities, and are determined using proportional controllers:

$$\begin{cases} \omega = \min(k_t(\theta_{des} - \theta_r), \omega_{\max}), \\ v = k_v v_{\max}(1 - \alpha \frac{|\omega|}{\omega_{\max}}) \end{cases} \tag{2}$$

where k_t, k_v, and α are constant proportional gains, θ_r is the current heading of the robot, and $\omega_{\max}$ and $v_{\max}$ are the maximum angular and linear velocities respectively.

C. Temple University

The team at Temple[6] used a deep reinforcement learning (DRL) based control policy, called DRL-VO [37], originally designed for safe and efficient navigation through crowded dynamic environments. The system architecture of the DRL-VO control policy, shown in Fig. 5, is divided into two modules: preprocessing and DRL network.

1) Preprocessing Module: Instead of directly feeding the raw sensor data into deep neural networks like other works [9], [19], [38], [39], the DRL-VO control policy utilizes preprocessed data as the network input. There are three types of inputs that capture different aspects of the scene.

1) **Pedestrians**: To track pedestrians, the raw RGB image data and point cloud data from a ZED camera are fed into the YOLOv3 [40] object detector to get pedestrian detections. These detections are passed into a multiple hypothesis tracker (MHT) [41] to estimate the number of pedestrians and their kinematics (i.e., position and velocity). These pedestrian kinematics are encoded into two 80×80 occupancy grid-style maps.
2) **Scene Geometry**: To track the geometry, the past 10 scans (0.5 s) of LiDAR data are collected. Each LiDAR scan is downsampled using a combination of minimum pooling and average pooling, and these downsampled LiDAR data are then reshaped and stacked to create an 80×80 LiDAR map.
3) **Goal Location**: To inform the robot where to go, the final goal point and its corresponding nominal path are fed into the pure pursuit algorithm [42] to extract the sub-goal point, which is fed into the DRL-VO network.

This novel preprocessed data representation is one key idea of the DRL-VO control policy, allowing it to bridge the sim-to-real gap and generalize to new scenarios better than other end-to-end policies.

2) DRL Network Module: The DRL-VO control policy uses an early fusion network architecture to combine the pedestrian and LiDAR data at the input layer in order to obtain high-level abstract feature maps. This early fusion architecture facilitates the design of small networks with fewer parameters than late fusion works [43], [44], which is the key deploying them on resource-constrained robots. These high-level feature maps are combined with the sub-goal point and

[6] https://sites.temple.edu/trail/

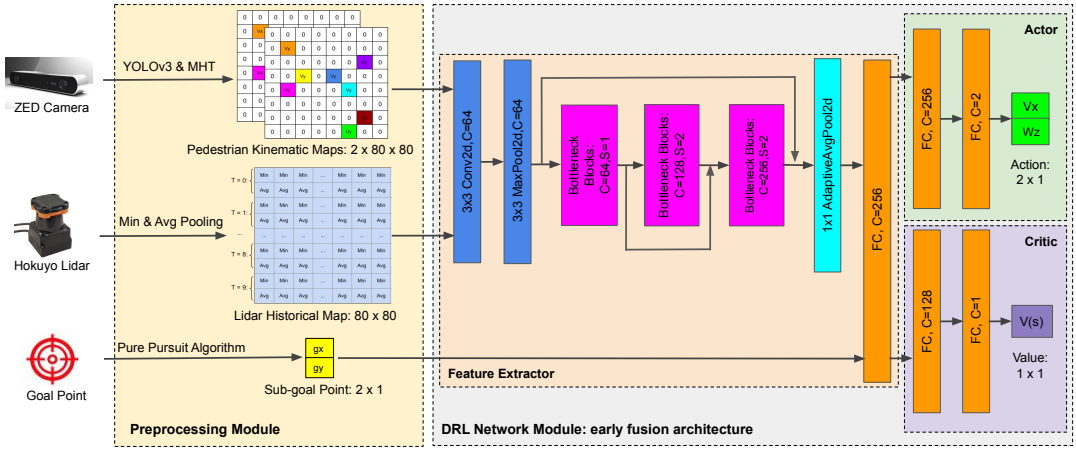


Fig. 5: (Temple Team) The system architecture of the DRL-VO control policy. Raw sensor data from the ZED camera and Hokuyo LiDAR, as well as the goal point, are fed into a preprocessing module to create intermediate data representations. These low-level intermediate features are fused in a feature extractor network to obtain high-level abstract features. The actor network uses these abstract features to generate steering commands to control the robot, while the critic network outputs the state value for training the policy.

fed into the actor and critic networks to generate suitable control inputs and a state value, respectively.

3) Training: To ensure that the DRL-VO policy leads the robot to navigate safely and efficiently, the team at Temple developed a new multi-objective reward function that rewards travel towards the goal, avoiding collisions with static objects, having a smooth path, and actively avoiding future collisions with pedestrians. This final term, which utilizes the concept of velocity obstacles (VO) [45], [46], is key to the success of the DRL-VO control policy. With this reward function, the DRL-VO policy is trained via the proximal policy optimization (PPO) algorithm [47] in a 3D lobby Gazebo simulation environment with 34 moving pedestrians using a Turtlebot2 robot with a ZED camera and a 2D Hokuyo LiDAR.

4) Deployment: The Temple team directly deployed the DRL-VO policy trained on a Turtlebot2 in The BARN Challenge without any model fine-tuning. To achieve this, the team had to account for three key differences:

1) **Unknown Map**: During development, the DRL-VO policy used a known occupancy grid map of the static environment for localization, which is not available in the BARN challenge. To account for this, the localization module (`amcl`) was removed from the software stack and replaced with a laser odometry module.
2) **Static Environment**: The DRL-VO policy was designed to function in dynamic environments. To account for the fact that the environments in the BARN Challenge were all static and highly constrained, the pedestrian map was set to all zeros.
3) **Different Robot Model**: The DRL-VO policy was trained on a Turtlebot2, which has a different maximum speed and footprint compared to the Jackal platform. In the BARN Challenge, the maximum speed of the robot was modified based on its proximity to obstacles. This led to the robot moving slowly (0.5 m/s, same speed as the Turtlebot2) when near obstacles and quickly (up to 2 m/s, maximum speed of the Jackal) when in an open area.

V. Discussions

Based on each team's approach and the navigation performance observed during the competition, we now discuss lessons learned from The BARN Challenge and point out promising future research directions to push the boundaries of efficient mobile robot navigation in highly constrained spaces.

A. Generalizability of Learning Based Systems

One surprising discrepancy between the simulation qualifier and the physical finals is the contrasting performance of the DRL-VO approach by Temple University, which outperformed all baselines and other participants in simulation, but suffers from frequent collision with obstacles in the real world. Despite the fact that the organizers modified the rules during the physical finals to allow the teams to make last-minute modifications to their navigation systems, DRL-VO still did not perform well in all three physical obstacle courses. The TRAIL team believes this is due to two types of *gap* between the simulator and the real world: 1) the real world environments were all highly constrained, whereas the simulator environments contained both constrained and unconstrained environments, and 2) the DRL-VO policy was learned on a Turtlebot2 model (which has a smaller physical footprint than a Jackal). Most of the collisions during the hardware tests were light grazes on the side, so a robot with a smaller size may have remained collision-free.

The stark performance contrast between simulation and the real world suggests a generalizability gap for the reinforcement learning approach. It was not practical for the team to re-train a new system on site during the competition,

given the impractically massive amount of training data required for reinforcement learning. How to address this generalizability gap and to make a navigation policy trained in simulation generalizable to the real world and different robot/sensor configurations remains to be investigated, even for such a simple, static obstacle avoidance problem.

Another potential way to address such inevitable generalizability gaps is to seek help from a secondary classical planner that identifies out-of-distribution scenarios in the real world and recovers from them through rule-based heuristics. In fact, for the last two physical courses, the Temple team tried to implement just such a "recovery planner" as a backup for DRL-VO: when a potential collision is detected (i.e., the robot faces an obstacle that is too close), the robot rotates in place to head towards an empty space in an attempt to better match the real-world distribution to that in the simulation during training. Although such a recovery planner did help in some scenarios, it is difficult for it to cover every difficult scenario and navigate through the entire obstacle course. Indeed, the Temple team spent time during the 30-minute timed sessions to fine tune the parameters of the recovery planner, but found it difficult to find a single set of parameters to recover the robot from all out-of-distribution scenarios while not to accidentally drive the robot into such scenarios throughout the entire course. On one hand, the simple nature of the recovery planner designed onsite during the competition contributed to the failure. On the other hand, tuning parameters of a planner to cover as many scenarios as possible remains a difficult problem, and will be discussed further below.

B. Tunability of Classical Systems

Similar to Temple's rule-based recovery planner, UT Austin team's entire navigation system relies on classical methods: EnML localization, medium-horizon kinematic planner, and local rollout-based kinodynamic planner. Inevitably, these classical approaches have numerous tuning parameters, which need to be correctly tuned to cover as many scenarios as possible. A natural disadvantage of relying on a single set of parameters to cover all different difficult scenarios in the BARN Challenge (e.g., dense obstacle fields, narrow curving hallways, relatively open spaces) is the inevitable tradeoff or compromise to sacrifice performance in some scenarios in order to succeed in others or to decrease speed for better safety. Indeed, the UT Austin team's strategy in the physical finals is to spend the first 20 minutes in the 30-minute timed session to fine tune the system parameters until a good set of parameters that allow successful navigation through the entire obstacle course is found, then finish three successful "safety trials" first, and finally re-tune the system to enable faster, more aggressive, but riskier navigation behaviors to reduce average traversal time. Although most such "speed trials" failed, luckily for the UT Austin team, other teams' inability to safely finish three collision-free trials to the goal make them the winner of the BARN Challenge only with a higher success rate (not faster navigation).

Two orthogonal future research directions can potentially help with the tunability of navigation systems: (1) developing planners free of tunable parameters onsite during deployment, such as end-to-end learning approaches, but, as mentioned above, with significantly better sim-to-real transfer and generalizability; (2) enabling more intelligent parameter tuning of classical systems, rather than laborious manual tuning, for example, through automatic tuning [48] or even dynamic parameter policies [49] learned from teleoperated demonstration [50], corrective interventions [51], evaluative feedback [52], or reinforcement learning [30].

C. Getting "Unstuck"

Although most of the failure trials during the physical finals were due to collision with obstacles, there were also many trials that did not succeed because the robot got stuck in some densely populated obstacle areas. In those places, the robot kept repeating the same behaviors multiple times, e.g., detecting imminent collision with obstacles, rotating in place, backing up, resuming navigation, detecting the same imminent collision again, and so on. Such behavior sometimes led to collision with an obstacle, sometimes got the robot stuck forever, and may also succeed in rare occasions. All three teams have experienced such behaviors, with the UT Austin and UVA teams being able to fix it by tuning parameters and the Temple team changing the threshold between DRL-VO and the recovery planner.

Similarly, in real-world autonomous robot navigation, how to get "unstuck" safely remains a common and challenging problem. No matter how intelligent an autonomous mobile robot is, it may still make mistakes in the real world, e.g., when facing scenarios out of the training distribution, corner cases not considered by the system developer, or situations where the current parameter set is not appropriate. It is very likely that the robot will repeat the same mistake over and over, e.g., getting stuck at the same place, which needs to be avoided. Future research should investigate ways to identify such "stuck" situations, balance the tradeoff between exploitation and exploration (i.e., when to keep trying the previous way vs. when to try out new ways to get unstuck), utilize previous successful exploratory experiences in future similar scenarios to not get stuck again [53], or leverage offline computation to correct such failure cases in the future [54].

D. Tradeoff between Safety and Speed

While The BARN Challenge was originally designed to test existing navigation system's speed of maneuvering through highly constrained obstacle environments, given the safety constraint of being collision-free, it ended up being a competition about safety alone. The UT Austin team won the competition simply by safely navigating eight out of nine physical trials, not by doing so with the fastest speed. All the teams, except the UT Austin team after they figured out an effective set of parameters for each physical obstacle course, struggled with simply reaching the goal without any

collision. The challenge organizers also deployed the widely-used DWA planner [5] in the ROS `move_base` navigation stack in the physical obstacle courses, only to find out that, despite being relatively safe compared to the participating teams' methods, it struggled with many narrow spaces and got stuck in those places very often. Such a fact shows that the current autonomous mobile robot navigation capability still lags farther behind than one may expect.

E. Latency Compensation for High Speed

Only the UT Austin team attempted to pursue higher speed navigation (> 0.5 m/s), doing so after an appropriate parameter set was found for the particular physical course and three successful "safety trials" have been achieved. However, most "speed trials" ended in collision. One contributing factor to such failure was improper latency compensation for various high speeds. The UT Austin team was the only team that explicitly considered latency compensation in their AMRL stack [32], through a latency parameter. During high-speed maneuvers, the robot inevitably needs to aggressively change its navigation speed to swerve through obstacles and to accelerate in open spaces. System latency caused by sensing, processing, computation, communication, and actuation will likely invalidate previously feasible plans. While simply tuning the latency parameter value can help to certain extent, a more intelligent and adaptive way to calculate and compensate system latency is necessary for the robot to take full advantage of its computing power before executing aggressive maneuvers.

F. Navigation is More Than Planning

To plan agile navigation maneuvers through highly constrained obstacle environments, the robot first needs to accurately perceive its configuration with respect to the obstacles. Inaccurate localization or odometry during fast maneuvers with significant angular velocity usually produces significant drift, causing previously valid plans become infeasible. While all three teams' local planners rely on raw perception to minimize such adverse effect, e.g., using high frequency laser scans and directly planning with respect to these raw features, their global planner usually depends on the results of localization or odometry techniques. For example, the Temple team used the Dijkstra's global planner in `move_base`. An erroneous localization will cause an erroneous global plan, which in turn will affect the quality of the local plan. Such adverse effect will diminish when the navigation speed is low, because localization techniques may recover from drift over time. During high-speed navigation, however, the planner needs to quickly plan actions regardless of whether the drift has been fixed or not. As mentioned above, latency will start to play a role as well, because a good latency compensation technique will depend on an accurate localization and odometry model of the robot, i.e., being able to predict where the robot will be based on where the robot is and what action will be executed. Techniques for better odometry, localization [33], and kinodynamic models [15], [55], [56] during high-speed navigation will be necessary to allow mobile robots to move both fast and accurately at the same time.

VI. CONCLUSIONS

The results of The BARN Challenge at ICRA 2022 suggest that, contrary to the perception of many in the field, autonomous metric ground robot navigation can not yet be considered a solved problem. Indeed, even the competition organizers had initially assumed that obstacle avoidance alone was too simple a goal, and therefore emphasized navigation speed before the physical competition. However, each of the finalist teams experienced difficulty performing collision-free navigation, and this ultimately led the organizers to modify the competition rules to focus more on collision avoidance. This result suggests that state-of-the-art navigation systems still suffer from suboptimal performance due to potentially many aspects of the full navigation system (discussed in Section V). Therefore, while it is worthwhile to extend navigation research in directions orthogonal to metric navigation (e.g., purely vision-based, off-road, and social navigation), the community should also not overlook the problems that still remain in this space, especially when robots are expected to be extensively and reliably deployed in the real world.

REFERENCES

[1] C. Rösmann, W. Feiten, T. Wösch, F. Hoffmann, and T. Bertram, "Trajectory modification considering dynamic constraints of autonomous robots," in *ROBOTIK 2012; 7th German Conference on Robotics*. VDE, 2012, pp. 1–6.

[2] ——, "Efficient trajectory optimization using a sparse model," in *2013 European Conference on Mobile Robots*. IEEE, 2013, pp. 138–143.

[3] C. Rösmann, F. Hoffmann, and T. Bertram, "Planning of multiple robot trajectories in distinctive topologies," in *2015 European Conference on Mobile Robots (ECMR)*. IEEE, 2015, pp. 1–6.

[4] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *[1993] Proceedings IEEE International Conference on Robotics and Automation*. IEEE, 1993, pp. 802–807.

[5] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[6] C. Rösmann, F. Hoffmann, and T. Bertram, "Integrated online trajectory planning and optimization in distinctive topologies," *Robotics and Autonomous Systems*, vol. 88, pp. 142–153, 2017.

[7] ——, "Kinodynamic trajectory optimization and control for car-like robots," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 5681–5686.

[8] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion planning and control for mobile robot navigation using machine learning: a survey," *Autonomous Robots*, pp. 1–29, 2022.

[9] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots," in *2017 ieee international conference on robotics and automation (icra)*. IEEE, 2017, pp. 1527–1533.

[10] 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.

[11] H. Karnan, G. Warnell, X. Xiao, and P. Stone, "Voila: Visual-observation-only imitation learning for autonomous navigation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 2497–2503.

[12] P. Atreya, H. Karnan, K. S. Sikand, X. Xiao, G. Warnell, S. Rabiee, P. Stone, and J. Biswas, "High-speed accurate robot control using learned forward kinodynamics and non-linear least squares optimization," in *To Appear in 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022.

[13] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. A. Theodorou, and B. Boots, "Imitation learning for agile autonomous driving," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 286–302, 2020.

[14] G. Kahn, P. Abbeel, and S. Levine, "Badgr: An autonomous self-supervised learning-based navigation system," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1312–1319, 2021.

[15] H. Karnan, K. S. Sikand, P. Atreya, S. Rabiee, X. Xiao, G. Warnell, P. Stone, and J. Biswas, "Vi-ikd: High-speed accurate off-road navigation using learned visual-inertial inverse kinodynamics," in *To Appear in 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022.

[16] R. Mirsky, X. Xiao, J. Hart, and P. Stone, "Prevention and resolution of conflicts in social navigation–a survey," *arXiv preprint arXiv:2106.12113*, 2021.

[17] C. Mavrogiannis, F. Baldini, A. Wang, D. Zhao, P. Trautman, A. Steinfeld, and J. Oh, "Core challenges of social robot navigation: A survey," *arXiv preprint arXiv:2103.05668*, 2021.

[18] 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*, 2022.

[19] P. Long, T. Fan, X. Liao, W. Liu, H. Zhang, and J. Pan, "Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6252–6259.

[20] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 285–292.

[21] "irobot – robot vacuum and mop," https://www.irobot.com/, accessed: 2022-07-20.

[22] "Home – everyday robots," https://everydayrobots.com/, accessed: 2022-07-20.

[23] "Meet scout," https://www.aboutamazon.com/news/transportation/meet-scout, accessed: 2022-07-20.

[24] "Starship," https://www.starship.xyz/, accessed: 2022-07-20.

[25] "The barn challenge," https://people.cs.gmu.edu/~xxiao2/Research/BARN_Challenge/BARN_Challenge.html, accessed: 2022-08-20.

[26] "Jackal ugv - small weatherproof robot - clearpath," https://clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/, accessed: 2022-07-21.

[27] D. Perille, A. Truong, X. Xiao, and P. Stone, "Benchmarking metric ground navigation," in *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2020, pp. 116–121.

[28] Z. Xu, B. Liu, X. Xiao, A. Nair, and P. Stone, "Benchmarking reinforcement learning techniques for autonomous navigation."

[29] Z. Wang, X. Xiao, A. J. Nettekoven, K. Umasankar, A. Singh, S. Bommakanti, U. Topcu, and P. Stone, "From agile ground to aerial navigation: Learning from learned hallucination," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 148–153.

[30] Z. Xu, G. Dhamankar, A. Nair, X. Xiao, G. Warnell, B. Liu, Z. Wang, and P. Stone, "Applr: Adaptive planner parameter learning from reinforcement," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 6086–6092.

[31] "ROS movebase navigation stack," http://wiki.ros.org/move_base, accessed: 2021-09-9.

[32] J. Biswas, "Amrl autonomy stack," https://github.com/ut-amrl/graph_navigation, 2013.

[33] "ROS movebase navigation stack," http://wiki.ros.org/move_base, accessed: 2021-09-9.

[34] N. Mohammad and N. Bezzo, "A robust and fast occlusion-based frontier method for autonomous navigation in unknown cluttered environments," in *To Appear in 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[35] M. Mujahad, D. Fischer, B. Mertsching, and H. Jaddu, "Closest gap based (cg) reactive obstacle avoidance navigation for highly cluttered environments," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 1805–1812.

[36] J. Minguez and L. Montano, "Nearness diagram (nd) navigation: collision avoidance in troublesome scenarios," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 1, pp. 45–59, 2004.

[37] Z. Xie and P. Dames, "Drl-vo: Using velocity obstacles to learn safe navigation policies for crowded dynamic scenes," *IEEE Transactions on Robotics*, 2022, under review.

[38] T. Fan, P. Long, W. Liu, and J. Pan, "Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios," *The International Journal of Robotics Research*, vol. 39, no. 7, pp. 856–892, 2020.

[39] R. Guldenring, M. Görner, N. Hendrich, N. J. Jacobsen, and J. Zhang, "Learning local planners for human-aware navigation in indoor environments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 6053–6060.

[40] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[41] K. Yoon, Y.-m. Song, and M. Jeon, "Multiple hypothesis tracking algorithm for multi-target multi-camera tracking with disjoint views," *IET Image Processing*, vol. 12, no. 7, pp. 1175–1184, 2018.

[42] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Carnegie-Mellon UNIV Pittsburgh PA Robotics INST, Tech. Rep., 1992.

[43] A. J. Sathyamoorthy, J. Liang, U. Patel, T. Guan, R. Chandra, and D. Manocha, "Densecavoid: Real-time navigation in dense crowds using anticipatory behaviors," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 11 345–11 352.

[44] X. Huang, H. Deng, W. Zhang, R. Song, and Y. Li, "Towards multi-modal perception-based navigation: A deep reinforcement learning method," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4986–4993, 2021.

[45] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[46] D. Wilkie, J. Van Den Berg, and D. Manocha, "Generalized velocity obstacles," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 5573–5578.

[47] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[48] H. Ma, J. S. Smith, and P. A. Vela, "Navtuner: Learning a scene-sensitive family of navigation policies," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 492–499.

[49] X. Xiao, Z. Wang, Z. Xu, B. Liu, G. Warnell, G. Dhamankar, A. Nair, and P. Stone, "Appl: Adaptive planner parameter learning," *Robotics and Autonomous Systems*, vol. 154, p. 104132, 2022.

[50] X. Xiao, B. Liu, G. Warnell, J. Fink, and P. Stone, "Appld: Adaptive planner parameter learning from demonstration," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4541–4547, 2020.

[51] Z. Wang, X. Xiao, B. Liu, G. Warnell, and P. Stone, "Appli: Adaptive planner parameter learning from interventions," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 6079–6085.

[52] Z. Wang, X. Xiao, G. Warnell, and P. Stone, "Apple: Adaptive planner parameter learning from evaluative feedback," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7744–7749, 2021.

[53] B. Liu, X. Xiao, and P. Stone, "A lifelong learning approach to mobile robot navigation," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1090–1096, 2021.

[54] Z. Xu, A. Nair, X. Xiao, and P. Stone, "Learning real-world autonomous navigation by self-supervised environment synthesis."

[55] X. Xiao, J. Biswas, and P. Stone, "Learning inverse kinodynamics for accurate high-speed off-road navigation on unstructured terrain," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 6054–6060, 2021.

[56] P. Atreya, H. Karnan, K. S. Sikand, X. Xiao, G. Warnell, S. Rabiee, P. Stone, and J. Biswas, "High-speed accurate robot control using learned forward kinodynamics and non-linear least squares optimization," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022.