Autonomous Mario Kart in the Wild

Lessons Learned From the Earth Rover Challenge at IROS 2024

By Xuesu Xiao, Jie Tan, Michael Cho, David Hsu, Dhruv Shah, Joanne Truong, Ted Xiao, Naoki Yokoyama, Wenhao Yu, Tingnan Zhang, Zhuo Xu, Santiago Pravisani, Niresh Dravin, and Mohammad Alshamsi

The Earth Rover Challenge (ERC) took place at the 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2024), in Abu Dhabi, United Arab Emirates (Figure 1). The aim of the challenge was to evaluate state-of-the-art autonomous ground navigation systems to move mobile robots through outdoor real-world environments with a set of low-cost onboard sensors [red–green–blue (RGB) cameras, inertia sensors, wheel encoders, and GPS] and offboard computation enabled by 4G communication. Specifically, the task was to navigate standardized four-wheeled differential-drive ground robots across the globe from predefined start locations to GPS goal locations. Three teams from across the world participated in the challenge. The competition results revealed insights into deploying autonomous mobile robots in the wild without expensive onboard sensors and computation as well as the engineering of the environments. In this article, we report the results and findings of the first ERC at IROS 2024, present the approaches used by three teams, and discuss lessons learned from the challenge to point out future research directions.

Digital Object Identifier 10.1109/MRA.2025.3565211
Date of current version: 14 June 2025

THE ERC

Autonomous mobile robot navigation has been a problem studied by the robotics community for decades [1], [2]. Equipped with expensive onboard sensors and computers (such as lidars and GPUs), existing navigation systems can move robots from one point to another without collisions, mostly in controlled lab environments [3], [4], [5], with some in real-world public spaces, potentially with highly engineered environments, maps, and features [6], [7], [8]. However, deploying cost-effective autonomous mobile robots with low-cost sensors, e.g., RGB cameras, inertial measurement units (IMUs), wheel encoders, and GPS, especially in unseen environments, still requires extra research and engineering effort. The ERC aims to tackle such a challenge by providing standardized low-cost mobile robot systems and offboard computation as well as 4G communication infrastructure to navigate a fleet of such affordable robots worldwide.

ERC OVERVIEW

The ERC took place as a conference competition at IROS 2024, in Abu Dhabi. As an urban robotic navigation competition, in the ERC, various research teams' autonomous navigation systems compete against top human gamers in a real-world "in the wild" setting. Both the artificial intelligence (AI) teams and human gamers

FIGURE 1. The first ERC, in Abu Dhabi.

are tasked to remotely control small-sized sidewalk robots deployed in various cities around the world and undertake predefined navigation missions with GPS location goals. To be specific, each team is given a four-wheeled differential-drive FrodoBot Zero robot to navigate around the globe, equipped with front and rear RGB cameras, an IMU, wheel encoders, and a GPS unit. Each navigation mission is given a difficulty ranking beforehand, based on factors such as the diversity of terrain and level of dynamism in the surroundings (e.g., cars, bicycles, and human pedestrians). Both the AI teams and human gamers aim to gain in-game points based on this difficulty ranking and their progress of mission completion for a given mission (measured by checkpoints reached versus the total number of checkpoints) within 1 h. In particular, the AI teams are given additional leniency, with up to three teleoperated interventions, although the total earned points are halved once an intervention has been used during a mission. Before the competition at IROS 2024, all the AI teams were provided with practice trials with robots in different cities worldwide and the FrodoBots-2K dataset [9]. The final eight cities, however, were not all seen during the practice trials and in the FrodoBots-2K dataset, as shown in Table 1.

TABLE 1. Cities seen (✓) versus unseen (✗) in the practice trials and FrodoBots-2K Dataset.

CITY	PRACTICE	DATASET
King'Ong'O (Kenya)	✓	✗
Kisumu (Kenya)	✓	✗
Liuzhou (China)	✓	✓
Wuhan (China)	✓	✓
Manila (Philippines)	✓	✓
Port Louis (Mauritius)	✓	✗
Singapore (Singapore)	✓	✓
Abu Dhabi	✗	✗

ERC RESULTS

Table 2 reports abbreviated final results achieved by seven human gamers and the three AI teams, i.e., Seoul National University (SNU), National University of Singapore (NUS), and the University of Texas at Austin (UT Austin). Overall, the human gamers were found to be drastically more proficient than the AI teams, with all seven human gamers ranking above the three AI teams. In fact, the lowest-ranked humans earned a final score of 36 points versus the top-ranked AI team earning a mere 15.2 points. For more details, refer to Table 3, with each of the entries denoting the difficulty level, successful checkpoints reached/total number of checkpoints in a mission, mission completion time, number of interventions (for the AI teams), and final points earned in that mission.

TABLE 2. The abbreviated ERC results.

RANKING	PLAYER	FINAL SCORE
1	Human 1 (masterchi)	42
2	Human 2 (gellyquin)	42
3	Human 3 (.swooshyy.)	42
4	Human 4 (fede14)	38
5	Human 5 (zionxstatic)	38
6	Human 6 (some1ne2220)	36
7	Human 7 (dnangel7343)	36
8	AI 1 (Seoul National University)	15.2
9	AI 2 (National University of Singapore)	13.7
10	AI 3 (University of Texas at Austin)	1.2

The competition had many scenarios where the AI teams significantly underperformed their human counterparts. For example, while many human players could easily get a bearing on their robot's whereabouts and its orientation, many AI teams struggled to localize their robot at the start of the mission, often being stuck for minutes before moving beyond a few meters away from the starting point of the mission. Furthermore, despite having the privilege of teleoperated intervention by the human AI team members, the robots still flipped over the edge of sidewalks on a few occasions, a mistake experienced human gamers rarely make. Finally, overreliance on GPS or IMU data, which could be highly inaccurate at times or slow to update, also caused some of the AI teams to overcompensate in their maneuvers or get confused about their robot's location. In contrast, experienced human gamers, relying on video streams, could quickly discern a robots' whereabouts by ignoring faulty or not up-to-date GPS information displayed on the map and successfully travel to the next checkpoint.

COMPETITION TEAMS AND APPROACHES

In this section, we report the approaches of the three AI teams.

The Three AI Teams

Seoul National University
Hyung-Suk Yoon, Ji-Sung Bae, E-In Son, Ji-Hoon Hwang, Dong-Wook Kim, Kun Park, Yeon-Kyu Lee, Jung-Tak Kim, and Seung-Woo Seo

National University of Singapore
Joel Loo, Zishuo Wang, Nielsen Cugito, Yuwei Zeng, and Tianle Shen

University of Texas at Austin
Arthur Zhang, Zichao Hu, Dongmyeong Lee, Taijing Chen, Michael Munje, Luisa Mao, Hochul Hwuang, Peter Stone, and Joydeep Biswas

SNU

OVERVIEW

The SNU team designed its autonomous navigation framework (SNUVIN) in a module-based manner, as depicted in Figure 2. For SNUVIN, a single front image, GPS data, and checkpoints come as input, and the action comes as output. There are three main modules to SNUVIN: costmap generation (CG), localization (LOC), and the action planner (AP).

CG

First, the CG module generates a costmap that represents the current environment around the robot. It is important for the robot to understand the environment in a 3D space both structurally and semantically. Therefore, SNUVIN conducts depth estimation to generate a 3D point cloud for structural analysis and semantic segmentation to assign semantic information to every point in the point cloud from the input image. Then, SNUVIN voxelizes the point cloud to create a 3D occupancy grid, and by projecting along the z-axis, a 2D grid is generated corresponding to the horizontal xy-plane. Afterward, to represent the surrounding environment information with several factors, such as slope and roughness, SNUVIN calculates the average and standard deviation of surface normals of each grid cell. In addition, the final grid cost is calculated by summing those various cost elements. Through these procedures, a 2D costmap that represents the structural and semantical information of the surrounding environment is generated.

LOC

Second, the LOC module estimates the current pose of the robot. Although the robot's position data are provided by GPS, the provision is at 1 Hz and is not suitable for real-time operation. Therefore, the SNU team designed a LOC module that estimates GPS values for positions where GPS signals are not available. The SNU team used ORB-SLAM3 [10], which is one of the widely used visual simultaneous localization and mapping (SLAM) algorithms, to estimate the pose of the robot. However, ORB-SLAM3 can estimate only the relative pose, while the target goals (checkpoints) are given in the form of global coordinates. Therefore, the SNU team matched the coordinates on the global level by adding the GPS data and the relative odometry from the visual odometry output.

AP

Finally, the AP module gets a costmap and pose from the CG and LOC modules, respectively, and computes the action to reach the goal. Additionally, a heading computation module is added to calculate the robot's target heading using only image and GPS data in a learning-based approach. This is intended

TABLE 3. The full results.

	KING'ONG'O	KISUMU	LIUZHOU	WUHAN	MANILA	PORT LOUIS	SINGAPORE	ABU DHABI	TOTAL
masterchi	L7, 14/14,	L4, 8/8,	L2, 4/4,	L6, 8/8,	L10, 10/10,	L3, 12/12,	L6, 8/8,	L4, 11/11,	**42**
	14:10, **7**	06:53, **4**	13:33, **2**	06:13, **6**	16:06, **10**	11:43, **3**	06:48, **6**	09:01, **4**	(01:24:27)
gellyquin	L7, 14/14,	L4, 8/8,	L2, 4/4,	L6, 8/8,	L10, 10/10,	L3, 12/12,	L6, 8/8,	L4, 11/11,	**42**
	14:13, **7**	07:24, **4**	13:31, **2**	05:59, **6**	15:30, **10**	11:37, **3**	10:44, **6**	07:55, **4**	(01:26:53)
.swooshyy.	L7, 14/14,	L4, 8/8,	L2, 4/4,	L6, 8/8,	L10, 10/10,	L3, 12/12,	L6, 8/8,	L4, 11/11,	**42**
	14:16, **7**	07:05, **4**	13:31, **2**	05:56, **6**	18:25, **10**	11:04, **3**	10:36, **6**	07:42, **4**	(01:28:35)
fede14	L7, 14/14,	L4, 8/8,	L2, 4/4,	L6, 8/8,	L10, 10/10,	L3, 12/12,	L6, 8/8,	L4, 9/11,	**38**
	14:39, **7**	07:16, **4**	14:07, **2**	06:53, **6**	15:42, **10**	11:02, **3**	10:44, **6**	09:45, **0**	(01:30:08)
zionxstatic	L7, 14/14,	L4, 8/8,	L2, 4/4,	L6, 8/8,	L10, 10/10,	L3, 12/12,	L6, 8/8,	L4, 10/11,	**38**
	14:40, **7**	07:20, **4**	13:26, **2**	06:17, **6**	21:17, **10**	10:56, **3**	07:43, **6**	11:46, **0**	(01:33:25)
some1ne2220	L7, 14/14,	L4, 8/8,	L2, 4/4,	L6, 8/8,	L10, 10/10,	L3, 12/12,	L6, 3/8,	L4, 11/11,	**36**
	14:13, **7**	07:02, **4**	13:21, **2**	06:01, **6**	16:31, **10**	11:02, **3**	04:00, **0**	08:48, **4**	(01:20:58)
dnangel7343	L7, 14/14,	L4, 8/8,	L2, 4/4,	L6, 8/8,	L10, 10/10,	L3, 12/12,	L6, 3/8,	L4, 11/11,	**36**
	15:49, **7**	06:58, **4**	13:48, **2**	05:59, **6**	16:05, **10**	10:58, **3**	03:51, **0**	09:00, **4**	(01:22:28)
SNU	L7, 4/14, 3,	L4, 7/8, 3,	L2, 3/4, 0,	L6, 8/8, 0,	L10, 4/10, 3,	L3, 8/12, 0,	L6, 0/10, 0,	L4, 5/11, 2,	**15.16**
	26:12, **1**	32:59, **1.75**	42:23, **1.5**	27:02, **6**	52:02, **2**	29:00, **2**	12:30, **0**	46:37, **0.91**	(04:28:45)
NUS	L7, 7/14, 3,	L4, 4/8, 3,	L2, 4/4, 0,	L6, 8/8, 2,	L10, 8/10, 3,	L3, 5/12, 3,	L6, 2/10, 0,	L4, 6/11, 3,	**13.66**
	84:02, **0.75**	19:24, **1**	39:40, **2**	24:36, **3**	51:49, **4**	26:19, **0.62**	14:07, **1.2**	54:50, **1.09**	(05:14:47)
UT Austin	L7, 0/14, 3,	L4, 2/8, 0,	L2, 0/4, 1,	L6, 1/8, 1,	L10, 0/10, 0,	L3, 1/12, 1,	L6, 2/10, 3,	L4, 1/11, 0,	**1.24**
	50:25, **0**	61:29, **0.5**	44:27, **0**	47:19, **0.38**	45:06, **0**	29:13, **0**	68:50, **0**	63:30, **0.36**	(06:50:19)

Each entry includes the difficulty level, successful checkpoints reached/total number of checkpoints in a mission, mission completion time, number of interventions (for AI teams), and final points earned in that mission.

to prevent the target heading angle calculation in the controller from being incorrect due to pose errors resulting from GPS noise or visual odometry. This pose error is difficult to solve with the SNUVIN framework alone. Therefore, the SNU team adopted a hybrid approach that solves the limitations of the rule-based approach with machine learning by creating a module based on transformers. The transformer model, in Figure 3, is trained with imitation learning from expert navigation demonstrations. It gets the front image and GPS data from teleoperation and the satellite image from Google Maps and computes the heading angle that the expert would most likely set in a given situation. Finally, considering the heading angle from the 2D costmap with the pose and the heading computation module, the controller computes the final action to the goal.

SUMMARY

The SNU team designed its navigation AI with SNUVIN, a hybrid rule-based and learning-based approach. The SNU team implemented SNUVIN with Robotic Operating System on a PC with an Intel i7 CPU and RTX 4070 Ti. SNUVIN is able to operate at 10 Hz. However, as the raw data from the teleoperation come in at 3 Hz, SNUVIN operated at 3 Hz during the competition.

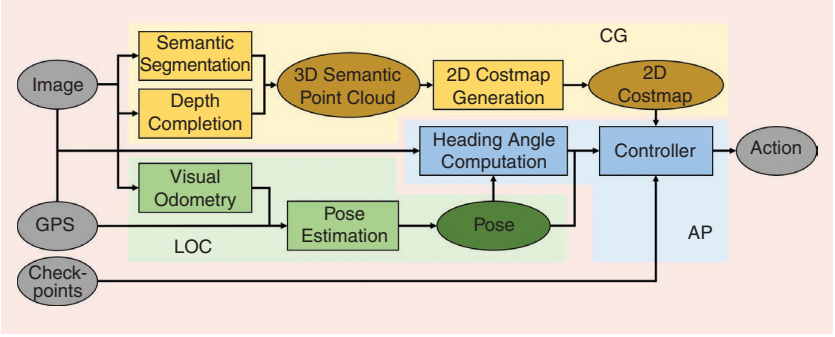

FIGURE 2. The architecture of SNUVIN. CG: costmap generation; LOC: localization; AP: action planner.

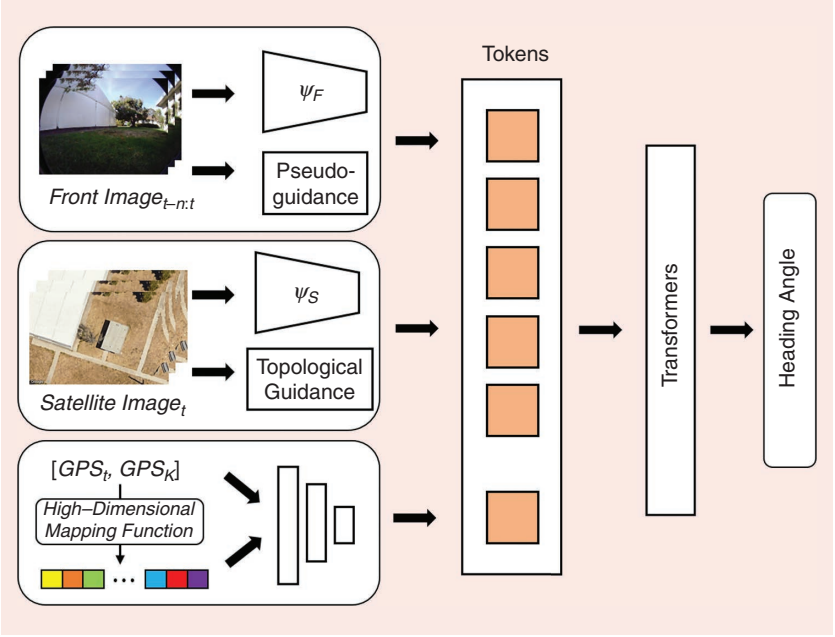

FIGURE 3. The architecture of the heading angle computation of the AP module. It is based on the transformer architecture. Tokens of front images are generated from the feature embedder ψ_F, and tokens of satellite images are generated from the feature embedder ψ_S. Finally, tokens of GPS data are generated and mapped to high dimensions to match the front image and satellite image tokens (the high-dimensional mapping function). This model is trained in a supervised manner from the FrodoBots-2K dataset.

NUS

OVERVIEW

The NUS team developed a modular system, addressing three key challenges underlying the task: 1) visual navigation with a monocular camera, 2) open-world natural human environments, and 3) low-frequency, high-latency sensing and control. Unreliable sensor streams coupled with noisy proprioception made accurate depth and scale estimation in the monocular setting challenging. To tackle visual navigation with a monocular camera, the choice was made to forgo 3D metric geometry estimation and focus instead on traversability estimation in 2D image space, relying on semantic image cues. To generalize over diverse scenes and appearance variations of open-world natural human environments, the system used visual features pretrained on large-scale datasets, with fine-tuning on selected portions of the FrodoBots-2K data.

Owing to hardware limitations and the unpredictability of latency, the low-frequency, high-latency sensing and control was harder to directly address. The system instead focused on handling navigation failures induced by suboptimal pathfinding and trajectory tracking, which arose from poor communication. This was achieved by augmenting the navigation pipeline with robust failure detection and recovery. These listed principles guided the system design. At a high level, the system (Figure 4) consists of perception, control, and failure detection and recovery modules. The perception module estimates traversability from RGB input and also issues an egocentric direction vector to the next checkpoint. The control module selects kinodynamically feasible trajectories aligned with the vector to the next checkpoint and generates control commands. The failure detection and recovery module is a supervisory monitor taking in raw RGB images and predicted traversability from perception to detect failures, overriding the control

module to execute heuristic recovery behaviors when necessary.

PERCEPTION

Given the need to operate in open-world human environments without reliable depth sensing due to the monocular setting, visual traversability prediction based on scene semantics was used. The perception module takes an RGB image as input and outputs a traversability mask based on the input image, with traversability scores in [0, 1]. Internally, a fast traversability estimator generates an initial mask, which is then further post-processed with clustering heuristics to identify and strongly penalize likely obstacles. The estimator uses pretrained DINO-ViT visual features, which enables strong generalization over diverse environments and allows for sample-efficient training and fine-tuning to adapt to new scenes.

To train an estimator for the wheeled FrodoBot configuration while capturing preferences on different terrains, a pipeline for automatically labeling data from FrodoBots-2K was developed. Based on the fixed egocentric camera view, the traversable region to which the robot is teleoperated is segmented with the Segment Anything Model [11], prompted with the bottom-central pixels. The Side Adapter Network [12] filters out low-quality images with motion blur and overexposure by checking and discarding images with no detected traversable areas.

TRAJECTORY GENERATION AND CONTROL

Kinodynamically feasible trajectories are chosen and tracked with a modified dynamic window approach (DWA) [2]. The DWA simplifies system design by unifying local planning and trajectory tracking since it generates trajectories parameterized with velocities to directly command the robot with. Its inputs are an egocentric heading toward the next subgoal and the 2D traversability mask, and it outputs linear and angular velocities, (v, ω). First, reactive obstacle avoidance is improved by modifying the DWA's search space to use more complex trajectory primitives. Trajectory primitives are extended from simple arcs to multisegmented arcs. Similar to model predictive control, each trajectory is rolled out for t_{sim} but followed only for $t_{track} < t_{sim}$. Second, trajectories are projected onto the traversability mask using camera intrinsics to evaluate kinematic feasibility in the absence of bird's-eye view (BEV) geometry information. A traversability score is summed from pixel values in the mask that lie within the trajectory inflated by the robot's projected footprint.

FAILURE DETECTION AND RECOVERY

The inevitability of failures in the open world is a key principle of the system's design, necessitating a module to recover from navigation failures. It monitors RGB inputs and traversability masks for failures, then activates heuristic recovery behaviors, which override the navigation layer to reset the robot. It maintains a severity level based on failure frequency, which balances between caution and the aggressiveness of corrective actions.

The module's strategy is to take successively more aggressive local actions to perturb the robot out of the failure state.

Two common failure modes are 1) suddenly encountering untraversable areas (e.g., when blocked by a dynamic obstacle) and 2) getting stuck in local minima (e.g., taking a wrong turn into a dead end). Detection of these modes is approximated by detecting overall low traversability across the mask and detecting that the robot is immobile despite being commanded to move. Upon failure detection, the module alternates between backtracking and perturbation behaviors. Backtracking executes cached actions open loop, while perturbations are local traversability-aware actions generated by the DWA with reduced goal weighting.

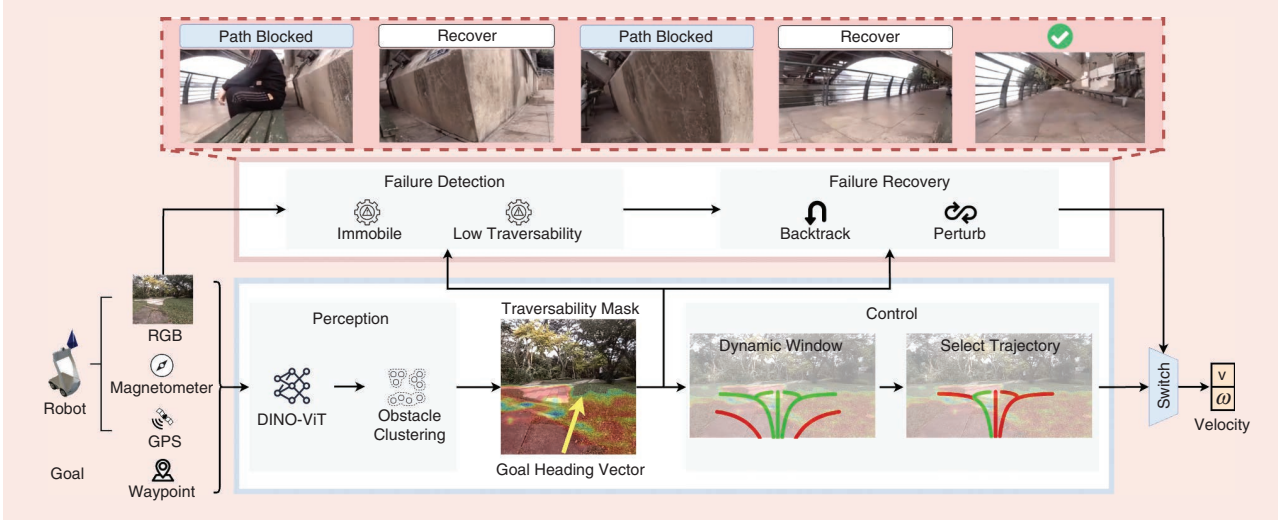

FIGURE 4. The system deals with purely monocular navigation across diverse locations via traversability estimation with pretrained models coupled with selection of kinodynamically feasible trajectories in image space, without explicit 3D geometry reconstruction. The open world and latency lead to inevitable failures, addressed by a high-level failure recovery system for monitoring and execution of heuristic recovery behaviors when necessary.

The magnitude of these actions increases with the severity level. Competition results empirically (Figure 4) demonstrate recovery to be crucial for escaping local minima in cluttered urban spaces (e.g., benches and bushes) and handling challenging areas with mixed terrains.

UT AUSTIN

SYSTEM DESIGN

The UT Austin team, Texas Trailblazers (T2), approached the challenge using a hybrid modular approach composed of the following modules (Figure 5):

- *Obstacle avoidance*: Hybrid module for geometric obstacle avoidance
- *Terrain preference alignment*: Learned module to prefer driving on specific terrains
- *Global LOC*: Classical module for localizing in the global map frame
- *Path planning*: Hybrid module for planning global paths and selecting viable local subgoals
- *Ackermann motion controller*: Classical motion controller for reaching local subgoals.

T2 utilizes service requests to confirm that the robot receives all outgoing commands before resuming the planning and control loop. This reduces the maximum operating frequency to guarantee that the planner does not execute on stale sensor observations.

OBSTACLE AVOIDANCE

The obstacle avoidance module generates a BEV obstacle costmap for motion planning and filtering out invalid local subgoals during a goal proposal stage. T2 used Metric3Dv2 [13], a monocular depth estimation model, and back projected to 3D to construct binary BEV costmaps.

TERRAIN PREFERENCE

T2 also employed PACER [14], a terrain-aware preference model, to predict a BEV terrain traversability costmap from RGB images. This approach avoids learning explicit classes of terrains by learning continuous embeddings through contrastive learning, improving generalization to environments with diverse terrains. PACER achieves generalization by training on a large dataset collected on the UT Austin campus along with synthetic terrain textures. Prior to deployment, operators can load in a preference context, i.e., a set of terrains with a preference order, to adjust the relative terrain costs without retraining.

LOC

T2 directly used GPS and magnetometer measurements to estimate the global pose. T2 assumed a Gaussian noise model for the GPS and found that this could adequately correct GPS measurement errors to localize to globally planned paths.

PLANNING AND CONTROLS

The planner plans in a global frame using a handcrafted traditional global planner. Once the waypoints are given, T2 first employs OpenStreetMap to plan a dense set of global goals to follow. The robot begins by rotating to align the goal GPS within its field of view. Once the goal is in view, the intermediate planner is triggered on demand. The planner uses RGB–depth images, obstacle costmaps, and terrain costmaps to determine local subgoals. Once the best local subgoal is selected, a motion planner selects the best path rollout to follow for a fixed time window (3 s). Motivated by recent success in using vision–language models (VLMs) for navigation, T2 blended VLM-based methods like PIVOT [15] and CONVOI [16] to select local navigation subgoals. Similar to PIVOT, BEV subgoal proposals are first generated, which are directly annotated on the image using number labels by projecting the 3D points to the pixel space using a projection matrix. Similar to CONVOI, T2 filters out a subgoal proposal if the subgoal corresponds to an area where an obstacle is. Furthermore, T2 filters out subgoals that require crossing multiple segmentation masks to reach, which was motivated by failure cases where the VLM would prefer subgoals that the robot could not navigate to, i.e., stairs. T2 used a similar text prompt to PIVOT, which was shown to work for

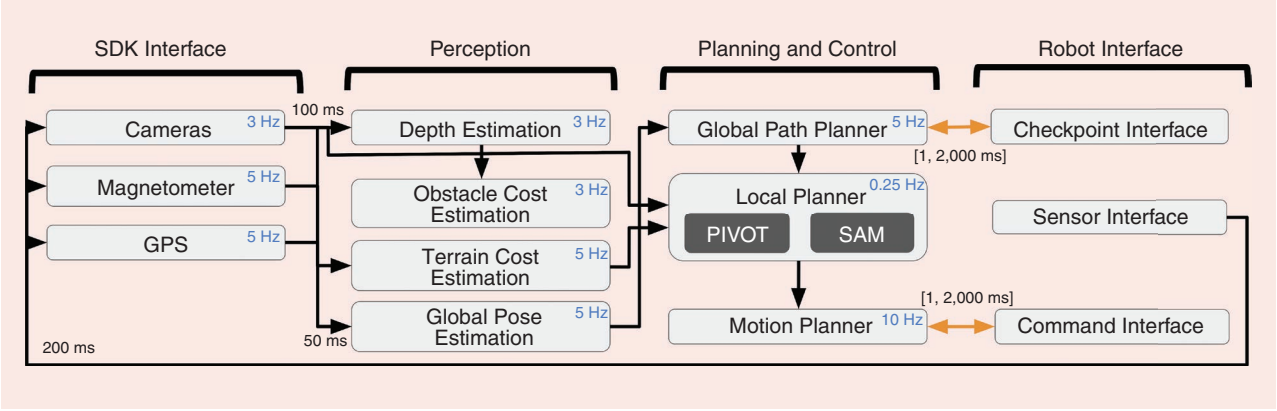

FIGURE 5. The T2 software system. The system is divided in to four main groups: remote software development kit (SDK) interface, perception, control, and robot interface. Average runtime frequencies for each module are indicated in blue, message latencies with a low standard deviation are indicated as a single number, and message latencies with a high standard deviation are indicated with a minimum and maximum latency range. Bidirectional service call communications are indicated in orange.

robot navigation. The VLM was GPT-4o mini, due to the need for a fast and capable model.

T2 used a handcrafted recovery policy when failing to identify valid local subgoals. Recovery begins with the robot rotating itself in place and scanning its surroundings to identify viable exploration targets by PIVOT. If a full rotation does not yield any valid goals, the robot attempts to move backward as an approximation to backtrack its past states.

DISCUSSIONS

Based on each team's approach and the navigation performance observed during the competition, we now discuss lessons learned from the first ERC and point out promising future research directions to push the boundaries of autonomous mobile robot navigation in the wild.

AI CANNOT COMPETE WITH HUMANS YET

The most prominent observation from the first ERC is that AI cannot compete with humans yet. In fact, all seven human players significantly outperformed the three AI teams, leaving a striking gap of 20.84 points between the last-place human player and the best AI team (whereas the difference between the first- and last-place human players was merely six points). As mentioned above, simple skills for humans, like state estimation, avoiding flipping over the edge of sidewalks, and distrusting unreliable sensor input, are still far from the reach of AI systems.

MODULAR APPROACH DOMINATES THIS COMPETITION

All three AI teams adopted modular approaches to the first ERC, instead of end-to-end learning [17], potentially due to a combination of insufficient training data and the complexity and dynamism of real-world navigation scenarios. We further observe the following two points across all three navigation systems:

- *The importance of a planner or controller*: All teams explicitly adopted a planner or controller, operating either in metric or image space, to produce the final actions to drive their robot. This stark contrast against the reported success of purely learning-based action generation methods in many academic papers showcases the crucial role of explicit planning and control and the importance to provide them with appropriate world representation in real-world navigation applications. Unlike simple lab spaces or controlled test courses used for academic research, the ERC's target domain is the real world in the wild, where out-of-distribution scenarios will be frequently encountered and cause problems for end-to-end learning methods trained only on a limited dataset.
- *Split over the necessity of explicit geometric representation for navigation*: SNU and UT Austin adopted explicit geometric representation in their modular systems using RGB-to-depth reconstruction in the form of 3D point clouds, 2D costmaps, and BEV maps. NUS explicitly avoided 3D metric geometry estimation and focused instead on traversability estimation in 2D image space, relying on semantic image cues. It is still an active debate whether explicit 3D geometric representation is necessary for navigation in the wild, especially considering the lack of expensive 3D lidars or depth cameras on affordable robot hardware.

> "UNLIKE SIMPLE LAB SPACES OR CONTROLLED TEST COURSES USED FOR ACADEMIC RESEARCH, THE ERC'S TARGET DOMAIN IS THE REAL WORLD IN THE WILD."

LEARNING IS AN ESSENTIAL COMPONENT FOR EACH TEAM

Despite the lack of end-to-end learning approaches in the first ERC, machine learning is still widely used in current modular systems, i.e., to learn a module, not the whole system. Such modules are mostly toward the perception side, including depth reconstruction, semantic segmentation, and feature extraction from RGB images as well as heading angle correction with the help of satellite images. However, classical approaches are preferred and used in the downstream planning and control tasks. The current practices and results of limiting the scope of learning to perception tasks show the promise of learning even with limited data and potentially reveal the lack of sufficient data to broaden the learning scope, e.g., learning action generation or learning end to end. Even when sufficient training data are available in the future, why and how learning should be applied to navigation will still need to be carefully considered by the robotics community [17].

FAILURE RECOVERY IS CRITICAL

Environments in the wild are full of unexpected scenarios, including blockage by dynamic obstacles or getting stuck in a dead end. Systems without error detection and handling may simply repeat the same erroneous action for an unlimited amount of time. Therefore, both recognizing such scenarios and driving the robot out of them are essential for long-duration and long-distance autonomous navigation tasks in the wild. All three teams, especially NUS, adopted specific error detection and handling techniques to recover from failures during the competition.

CHALLENGES OF LOW-COST MOBILE ROBOTS IN THE WILD

One unique feature of the ERC is its adoption of low-cost mobile robots to navigate the world. Such a feature is expected to raise challenges in terms of primitive low-quality perceptual streams as well as latencies caused by the need to off-load onboard computation to remote servers. While the latter has been addressed by the FrodoBot and three AI teams' engineering effort to optimize and account for latencies in

their systems, problems due to the low-cost sensors, especially when being complicated by a fleet of robots, have been reported by the teams, as follows:

- *Cross-robot differences*: Precise sensor calibration on each robot is required by many classical systems, like visual SLAM. For example, ORB-SLAM2 [18] and VINS-mono [19] estimate robot pose by minimizing the projection errors using the 3D map points estimated with the intrinsic parameters, while DPVO [20] depends on intrinsic parameters to project patches from the previous frames to the incoming frame using the estimated pose and depth. To resolve scale ambiguity in visual odometry, sensor fusion with an IMU or wheel encoder is necessary and requires precise calibration for each deployment. While calibration data are provided in the FrodoBots-2K dataset, cross-robot differences in sensor parameters introduce noises to the calibration and then, e.g., cause the odometry system to lose the track or depth or the map reconstruction to become imprecise. To address a fleet of low-cost robots with inevitable differences, potential future solutions include online calibration quality monitoring and recalibration techniques that do not require a dedicated calibration procedure [21], such as utilizing structure from motion to dynamically determine camera intrinsic parameters during initialization.
- *Unreliable GPS*: As observed in the challenge, GPS quality varies across the globe and is of particularly low quality for robots located in, e.g., Abu Dhabi. Without real-time kinematic fixation, blindly trusting unreliable GPS will significantly jeopardize the LOC and odometry, causing trouble for planning and control. Interestingly, such a problem has also caused trouble for some human players in determining robots' whereabouts and therefore where to drive, while other human players know when to distrust compromised GPS information. How to deal with noisy GPS of different qualities in different places for accurate state estimation remains a challenge for autonomous navigation systems.

AUTHORS

Xuesu Xiao, George Mason University, Fairfax, VA 22030 USA. E-mail: xiao@gmu.edu.

Jie Tan, Google DeepMind, Mountain View, CA 94043 USA. E-mail: jietan@google.com.

Michael Cho, FrodoBots Lab, Singapore 521163, Singapore.

David Hsu, National University of Singapore, Singapore 119077, Singapore.

Dhruv Shah, Google DeepMind, Mountain View, CA 94043 USA.

Joanne Truong, Meta, Menlo Park, CA 94025 USA.

Ted Xiao, Google DeepMind., Mountain View, CA 94043 USA.

Naoki Yokoyama, Georgia Institute of Technology, Atlanta, GA 30332 USA. E-mail: nyokoyama@gatech.edu

Wenhao Yu, Google DeepMind, Mountain View, CA 94043 USA.

Tingnan Zhang, Google DeepMind, Mountain View, CA 94043 USA.

Zhuo Xu, Google DeepMind, Mountain View, CA 94043 USA.

Santiago Pravisani, FrodoBots Lab, Singapore 521163, Singapore.

Niresh Dravin, FrodoBots Lab, Singapore 521163, Singapore.

Mohammad Alshamsi, Robotics and Automation Society, United Arab Emirates, 20000 Abu Dhabi, UAE.

REFERENCES

[1] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *Proc. IEEE Int. Conf. Robot. Automat.*, Piscataway, NJ, USA: IEEE Press, 1993, pp. 802–807, doi: 10.1109/ROBOT.1993.291936.

[2] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Autom. Mag.*, vol. 4, no. 1, pp. 23–33, Mar. 1997, doi: 10.1109/100.580977.

[3] X. Xiao et al., "Autonomous ground navigation in highly constrained spaces: Lessons learned from the benchmark autonomous robot navigation challenge at ICRA 2022 [Competitions]," *IEEE Robot. Autom. Mag.*, vol. 29, no. 4, pp. 148–156, 2022, doi: 10.1109/MRA.2022.3213466.

[4] X. Xiao et al., "Autonomous ground navigation in highly constrained spaces: Lessons learned from the second BARN challenge at ICRA 2023 [Competitions]," *IEEE Robot. Autom. Mag.*, vol. 30, no. 4, pp. 91–97, Dec. 2023, doi: 10.1109/MRA.2023.3322920.

[5] X. Xiao et al., "Autonomous ground navigation in highly constrained spaces: Lessons learned from the third BARN challenge at ICRA 2024 [Competitions]," *IEEE Robot. Autom. Mag.*, vol. 31, no. 3, pp. 197–204, Sep. 2024, doi: 10.1109/MRA.2024.3427891.

[6] "Home - Starship Technologies: Autonomous robot delivery." Starship Technologies, 2024. Accessed: May 6, 2025. [Online]. Available: https://www.starship.xyz/

[7] "Rent a delivery robot – Fast & green with Kiwibot." Kiwibot, 2024. Accessed: May 6, 2025. [Online]. Available: https://www.kiwibot.com/

[8] "Tiny Mile: Robot delivery." PinkBot - Affordable Miami Food Delivery, 2024. Accessed: May 6, 2025. [Online]. Available: https://tinymile.ai/

[9] "FrodoBots 2K dataset." FrodoBots, 2024. Accessed: May 6, 2025. [Online]. Available: https://huggingface.co/datasets/frodobots/FrodoBots-2K

[10] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021, doi: 10.1109/TRO.2021.3075644.

[11] A. Kirillov et al., "Segment anything," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 3992–4003, doi: 10.1109/ICCV51070.2023.00371.

[12] M. Xu, Z. Zhang, F. Wei, H. Hu, and X. Bai, "Side adapter network for open-vocabulary semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2023, pp. 2945–2954, doi: 10.1109/CVPR52729.2023.00288.

[13] M. Hu et al., "Metric3D v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 12, pp. 10,579–10,596, Dec. 2024, doi: 10.1109/TPAMI.2024.3444912.

[14] L. Mao, G. Warnell, P. Stone, and J. Biswas, "Pacer: Preference-conditioned all-terrain costmap generation," *IEEE Trans. Robot. Autom.*, vol. 10, no. 5, pp. 4572–4579, May 2025, doi: 10.1109/LRA.2025.3549645.

[15] S. Nasiriany et al., "PIVOT: Iterative visual prompting elicits actionable knowledge for VLMs," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 37,321–37,341.

[16] A. J. Sathyamoorthy et al., "CoNVOI: Context-aware navigation using vision language models in outdoor and indoor environments," 2024, *arXiv:2403.15637*.

[17] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion planning and control for mobile robot navigation using machine learning: A survey," *Auton. Robots*, vol. 46, no. 5, pp. 569–597, 2022, doi: 10.1007/s10514-022-10039-8.

[18] R. Mur-Artal and J. D. Tardo's, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017, doi: 10.1109/TRO.2017.2705103.

[19] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018, doi: 10.1109/TRO.2018.2853729.

[20] Z. Teed, L. Lipson, and J. Deng, "Deep patch visual odometry," *Advances Neural Inf. Process. Syst.*, vol. 36, pp. 39,033–39,051, 2024.

[21] X. Xiao, Y. Zhang, H. Li, H. Wang, and B. Li, "Camera-IMU extrinsic calibration quality monitoring for autonomous ground vehicles," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 4614–4621, Apr. 2022, doi: 10.1109/LRA.2022.3151970.