Autonomous Mario Kart in the Wild: Lessons Learned from The Earth Rover Challenge at IROS 2024

Organizers: Xuesu Xiao[1], Jie Tan[2], Michael Cho[3], David Hsu[4], Dhruv Shah[2], Joanne Truong[5], Ted Xiao[2], Naoki Yokoyama[6], Wenhao Yu[2], Tingnan Zhang[2], Zhuo Xu[2], Santiago Pravisani[3], Niresh Dravin[3], and Mohammad Alshamsi[7]
Teams: Hyung-Suk Yoon[8], Ji-Sung Bae[8], E-In Son[8], Ji-Hoon Hwang[8], Dong-Wook Kim[8], Kun Park[8], Yeon-Kyu Lee[8], Jung-Tak Kim[8], Seung-Woo Seo[8], Joel Loo[9], Zishuo Wang[9], Nielsen Cugito[9], Yuwei Zeng[9], Tianle Shen[9], Arthur Zhang[10], Zichao Hu[10], Dongmyeong Lee[10], Taijing Chen[10], Michael Munje[10], Luisa Mao[10], Hochul Hwuang[10], Peter Stone[10], and Joydeep Biswas[10]

***Abstract*— 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. The aim of the challenge was to evaluate state-of-the-art autonomous ground navigation systems to move mobile robots through outdoor, real-world enviornments with a set of low-cost onboard sensors (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 engineering of the environments. In this article, we report the results and findings of the 1st ERC at IROS 2024, present the approaches used by the three teams, and discuss lessons learned from the challenge to point out future research directions.**

I. The Earth Rover Challenge

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]–[5], some in real-world public spaces, potentially with highly engineered environment, maps, and features [6]–[8]. However, deploying cost-effective autonomous mobile robots with low-cost sensors, e.g., RGB cameras, Inertia Measurement Units (IMUs), wheel encoders, and Global Positioning System (GPS), especially in unseen environments, still requires extra research and engineering effort. The Earth Rover Challenge (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.

[1]George Mason University [2]Google DeepMind [3]FrodoBots [4]National University of Singapore [5]Meta [6]Georgia Institute of Technology [7]Robotics and Automation Society UAE [8]Seoul National University [9]National University of Singapore [10]The University of Texas at Austin

Fig. 1: The 1st Earth Rover Challenge in Abu Dhabi.

A. ERC Overview

ERC took place as a conference competition at the 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2024) in Abu Dhabi, United Arab Emirates. As an urban robotic navigation competition, various research teams' autonomous navigation systems compete against top human gamers in a real-world "in the wild" setting. Both AI teams or human gamers are tasked to remotely control small-sized sidewalk robots deployed in various cities around the world and undertake pre-defined navigation missions with GPS location goals. To be specific, each team is given a four-wheeled, differential-drive FrodoBot Zero robot 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 diversity of terrain and level of dynamism in the surroundings (e.g., cars, bicycles, and human pedestrians). Both AI teams or 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 vs total number of checkpoints) within one hour. In particular, AI teams are given additional leniency with up to 3 tele-operated interventions although the total earned points will be halved once an intervention had been used during a mission. Before the competition at IROS 2024 in Abu Dhabi, all AI teams are provided with practice trials with robots in different cities worldwide and the FrodoBots-

2K dataset [9]. The final eight cities, however, are not all seen during practice trials and in the FrodoBots-2K dataset, as shown in Table I.

TABLE I: Cities Seen (✓) vs. Unseen (✗) in Practice Trials and FrodoBots-2K Dataset.

Cities	Practice	Dataset
King'Ong'O (Kenya)	✓	✗
Kisumu (Kenya)	✓	✗
Liuzhou (China)	✓	✓
Wuhan (China)	✓	✓
Manila (Philippines)	✓	✓
Port Louis (Mauritius)	✓	✗
Singapore (Singapore)	✓	✓
Abu Dhabi (UAE)	✗	✗

B. ERC Results

Table II shows the abbreviated table of the final results achieved by seven human gamers and three AI teams, i.e., Seoul National University (SNU), National University of Singapore (NUS), and The University of Texas at Austin (UT Austin). Overall, human gamers are found to be drastically more proficient than the AI teams, with all 7 human gamers ranking above the 3 AI teams. In fact, the lowest ranked humans earned a final score of 36.0 points, versus the top ranked AI earning a mere 15.2 points. For more details, refer to Table III, with each of the table entry denoting 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.

TABLE II: Abbreviated ERC Results.

Ranking	Players	Final Score
1	Human #1 (masterchi_)	42.0
2	Human #2 (gellyquin)	42.0
3	Human #3 (.swooshyy.)	42.0
4	Human #4 (fede14)	38.0
5	Human #5 (zionxstatic)	38.0
6	Human #6 (some1ne2220)	36.0
7	Human #7 (dnangel7343)	36.0
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 witnessed many scenarios, where the AI teams significantly underperformed their human counterparts. For example, while many human players can easily get a bearing of the robot's whereabouts and its orientation, many AI teams struggled to localize the 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 tele-operated intervention by the human AI team members, the robots still flipped over the edge of the sidewalk in a few occasions, a mistake experienced human gamers will rarely make. Finally, over-reliance on GPS or IMU data, which could be highly inaccurate at times or slow to update, also caused some of the AI teams to over-compensate in their maneuvers or get confused of the robots' location. In contrast, experienced human gamers, relying on video streams, can quickly discern the robots' whereabouts by ignoring the faulty or not up-to-date GPS information displayed on the map and successfully travel to next checkpoints.

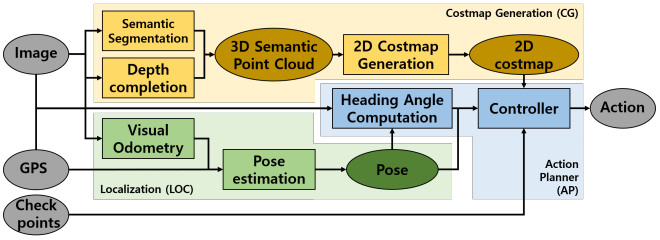

Fig. 2: Architecture of SNUVIN.

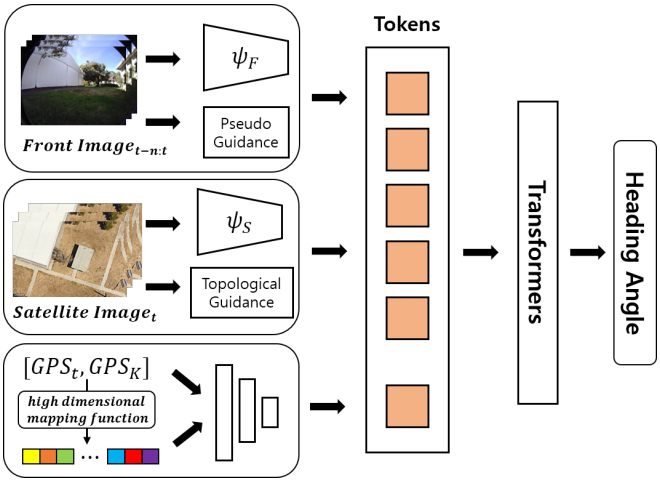

Fig. 3: Architecture of heading angle computation of the Action Planner module. It is based on the Transformers architecture. Tokens of front images are generated from the feature embedder ψ_F, and tokens of satellite image are generated from the feature embedder ψ_S. At last, the tokens of GPS data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

II. Competition Teams and Approaches

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

A. Seoul National University (SNU)

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

Costmap Generation. Firstly, the costmap generation (CG) module generates the costmap that represents the current environments around the robot. It is important for

TABLE III: Full Results. Each entry includes 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.

	King'Ong'O	Kisumu	Liuzhou	Wuhan	Manila	Port Louis	Singapore	Abu Dhabi	Total
masterchi_	L7, 14/14, 14:10, **7.0**	L4, 8/8, 06:53, **4.0**	L2, 4/4, 13:33, **2.0**	L6, 8/8, 06:13, **6.0**	L10, 10/10, 16:06, **10.0**	L3, 12/12, 11:43, **3.0**	L6, 8/8, 06:48, **6.0**	L4, 11/11, 09:01, **4.0**	**42.00** (01:24:27)
gellyquin	L7, 14/14, 14:13, **7.0**	L4, 8/8, 07:24, **4.0**	L2, 4/4, 13:31, **2.0**	L6, 8/8, 05:59, **6.0**	L10, 10/10, 15:30, **10.0**	L3, 12/12, 11:37, **3.0**	L6, 8/8, 10:44, **6.0**	L4, 11/11, 07:55, **4.0**	**42.00** (01:26:53)
.swooshyy.	L7, 14/14, 14:16, **7.0**	L4, 8/8, 07:05, **4.0**	L2, 4/4, 13:31, **2.0**	L6, 8/8, 05:56, **6.0**	L10, 10/10, 18:25, **10.0**	L3, 12/12, 11:04, **3.0**	L6, 8/8, 10:36, **6.0**	L4, 11/11, 07:42, **4.0**	**42.00** (01:28:35)
fede14	L7, 14/14, 14:39, **7.0**	L4, 8/8, 07:16, **4.0**	L2, 4/4, 14:07, **2.0**	L6, 8/8, 06:53, **6.0**	L10, 10/10, 15:42, **10.0**	L3, 12/12, 11:02, **3.0**	L6, 8/8, 10:44, **6.0**	L4, 9/11, 09:45, **0.0**	**38.00** (01:30:08)
zionxstatic	L7, 14/14, 14:40, **7.0**	L4, 8/8, 07:20, **4.0**	L2, 4/4, 13:26, **2.0**	L6, 8/8, 06:17, **6.0**	L10, 10/10, 21:17, **10.0**	L3, 12/12, 10:56, **3.0**	L6, 8/8, 07:43, **6.0**	L4, 10/11, 11:46, **0.0**	**38.00** (01:33:25)
some1ne2220	L7, 14/14, 14:13, **7.0**	L4, 8/8, 07:02, **4.0**	L2, 4/4, 13:21, **2.0**	L6, 8/8, 06:01, **6.0**	L10, 10/10, 16:31, **10.0**	L3, 12/12, 11:02, **3.0**	L6, 3/8, 04:00, **0.0**	L4, 11/11, 08:48, **4.0**	**36.00** (01:20:58)
dnangel7343	L7, 14/14, 15:49, **7.0**	L4, 8/8, 06:58, **4.0**	L2, 4/4, 13:48, **2.0**	L6, 8/8, 05:59, **6.0**	L10, 10/10, 16:05, **10.0**	L3, 12/12, 10:58, **3.0**	L6, 3/8, 03:51, **0.0**	L4, 11/11, 09:00, **4.0**	**36.00** (01:22:28)
SNU	L7, 4/14, 3, 26:12, **1.0**	L4, 7/8, 3, 32:59, **1.75**	L2, 3/4, 0, 42:23, **1.5**	L6, 8/8, 0, 27:02, **6.0**	L10, 4/10, 3, 52:02, **2.0**	L3, 8/12, 0, 29:00, **2.0**	L6, 0/10, 0, 12:30, **0.0**	L4, 5/11, 2, 46:37, **0.91**	**15.16** (04:28:45)
NUS	L7, 7/14, 3, 84:02, **0.75**	L4, 4/8, 3, 19:24, **1.0**	L2, 4/4, 0, 39:40, **2.0**	L6, 8/8, 2, 24:36, **3.0**	L10, 8/10, 3, 51:49, **4.0**	L3, 5/12, 3, 26:19, **0.62**	L6, 2/10, 0, 14:07, **1.2**	L4, 6/11, 3, 54:50, **1.09**	**13.66** (05:14:47)
UT Austin	L7, 0/14, 3, 50:25, **0.0**	L4, 2/8, 0, 61:29, **0.5**	L2, 0/4, 1, 44:27, **0.0**	L6, 1/8, 1, 47:19, **0.38**	L10, 0/10, 0, 45:06, **0.0**	L3, 1/12, 1, 29:13, **0.0**	L6, 2/10, 3, 68:50, **0.0**	L4, 1/11, 0, 63:30, **0.36**	**1.24** (06:50:19)

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 x-y 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.

Localization. Secondly, the localization (LOC) module estimates the current pose of the robot. Although the robot's position data is provided by GPS, it is at 1Hz 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 the ORB-SLAM3 [10], which is one of the widely used visual SLAM algorithms, to estimate the pose of the robot. However, ORB-SLAM3 can only estimate 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.

Action Planner. At last, the action planner (AP) module gets a costmap and pose from the CG and LOC module, 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 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. Their Transformers model in Fig. 3 is trained with imitation learning from expert navigation demonstrations. It gets the front image and GPS data from teleoperation and satellite image from Google map and computes the heading angle that the expert will most likely set in a given situation. Finally, considering the heading angle from the 2D costmap with pose, and the heading computation module, the controller computes the final action to the goal.

Summary. The SNU team designed their navigation AI with SNUVIN, a hybrid rule-based and learning-based approach. The SNU team implemented SNUVIN with ROS on a PC with Intel i7 CPU and RTX 4070 ti. SNUVIN is able to operate at 10 Hz. However, as the raw data from the teleoperation comes in at 3 Hz, SNUVIN operated at 3 Hz during the competition.

B. National University of Singapore (NUS)

Overview. The NUS team developed a modular system, addressing three key challenges underlying the task: **(i)** visual navigation with a monocular camera, **(ii)** open-world natural human environments, **(iii)** 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 **(i)**, 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 the diverse scenes and appearance variations of **(ii)**, the system used visual features pretrained on large-scale datasets, with fine-tuning on selected portions of the FrodoBots-2K data.

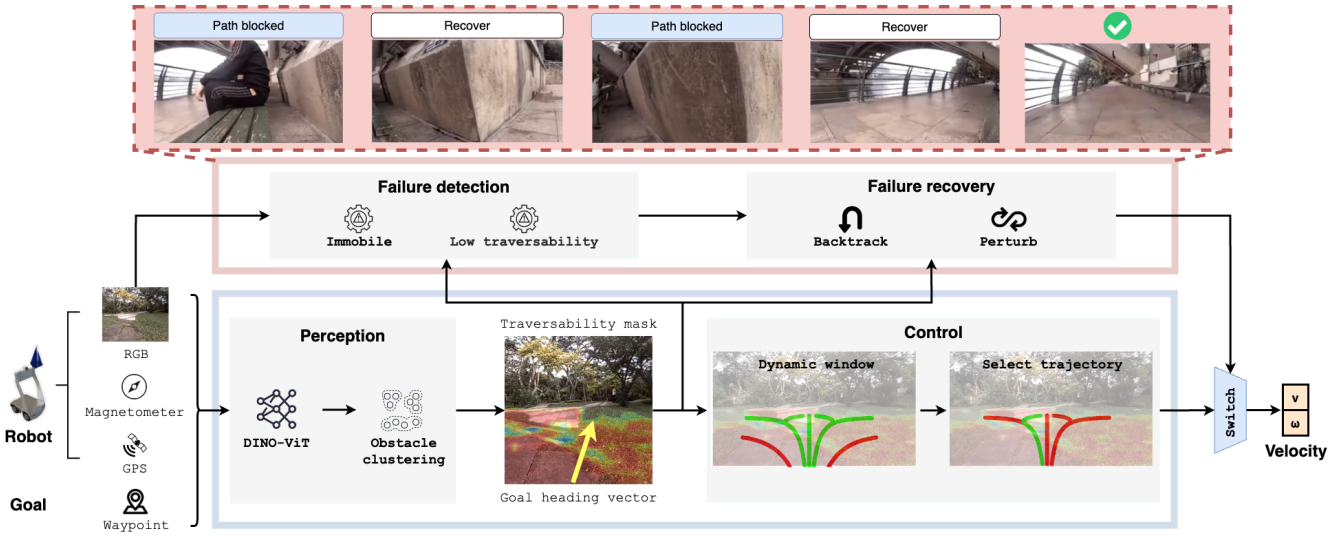

Fig. 4: The system deals with purely monocular navigation across diverse locations via traversability estimation with pretrained models coupled with selection of kinodynamically feasible trajectories in image space, without explicit 3D geometry reconstruction. Open-worldness and latency lead to inevitable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

Owing to hardware limitations and the unpredictability of latency, **(iii)** was harder to directly address. The system instead focused on handling navigation failures induced by suboptimal path-finding and trajectory-tracking, which arose from the poor communication. This was achieved by augmenting the navigation pipeline with robust failure detection and recovery. These listed principles guided system design.

At a high level, the system (Fig. 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 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 a 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 enable 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 region where the robot is currently teleoperated onto as traversable region 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]. 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). Firstly, reactive obstacle avoidance is improved by modifying DWA's search space to use more complex trajectory primitives. Trajectory primitives are extended from simple arcs to multi-segmented arcs. Similar to Model Predictive Control, each trajectory is rolled out for t_{sim} but only followed for $t_{track} < t_{sim}$. Secondly, trajectories are projected onto the traversability mask using camera intrinsics, to evaluate kinematic feasibility in the absence of bird's-eye view 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 aggressiveness of corrective actions.

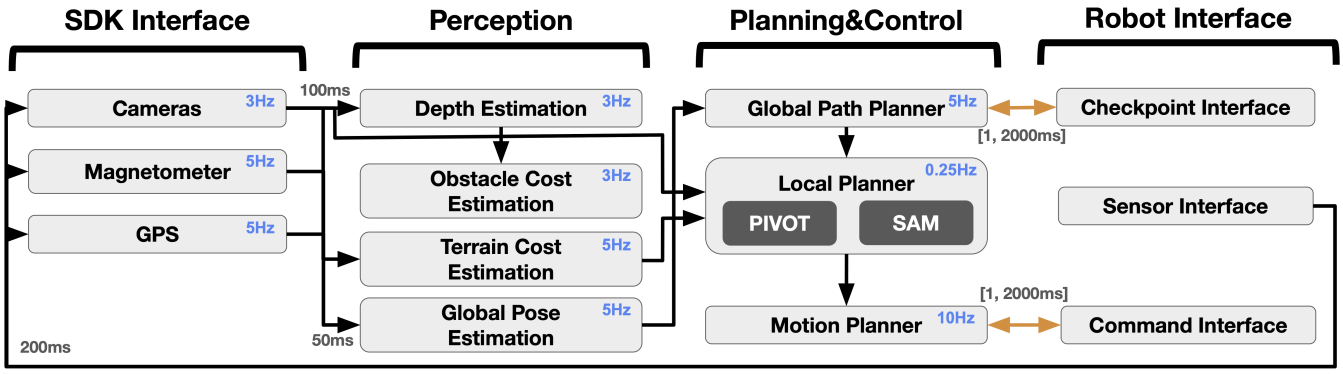

Fig. 5: Flowchart of Texas Trailblazers (T2) software system. The system is divided in to four main groups: Remote SDK interface, perception, control, and robot interface. Average runtime frequencies for each module are indicated in blue, message latencies with low standard deviation are indicated as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

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: **(i)** suddenly encountering untraversable areas (e.g., when blocked by a dynamic obstacle); **(ii)** 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 among *backtracking* and *perturbation* behaviors. Backtracking executes cached actions open-loop, while perturbations are local traversability-aware actions generated by DWA with reduced goal weighting. The magnitude of these actions increases with severity level. Competition results empirically (Fig. 4) show recovery to be crucial for escaping local minima in cluttered urban spaces (e.g., benches and bushes) and handling challenging areas with mixed terrains.

C. The University of Texas at Austin (UT Austin)

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

- **Obstacle Avoidance.** Hybrid module for geometric obstacle avoidance.
- **Terrain Preference Alignment.** Learned module to prefer driving on specific terrains.
- **Global Localization.** 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 utilized service requests to confirm that robots receive all outgoing commands before resuming the planning and control loop. This reduces the max operating frequency to guarantee the planner does not execute on stale sensor observations.

Obstacle Avoidance. The obstacle avoidance module generates a Bird's Eye View (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 backprojected 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 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.

Localization. T2 directly used GPS and magnetometer measurements to estimate global pose. T2 assumed a Gaussian noise model for the GPS and found this can 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 employed 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-D images, obstacle cost maps, 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 (3s).

Motivated by recent success in using 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 filtered out a subgoal proposal if the subgoal corresponds to an area where an obstacle is. Furthermore, T2 filtered 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 cannot navigate to, i.e., stairs. T2 used a similar text prompt to PIVOT, which was shown to work for robot navigation. The VLM was GPT-4o-mini, due to the need for a fast, 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 would attempt to move backward as an approximation to backtrack its past states.

III. Discussions

Based on each team's approach and the navigation performance observed during the competition, we now discuss lessons learned from the 1st ERC and point out promising future research directions to push the boundaries of autonomous mobile robot navigation in the wild.

A. AI cannot compete with humans yet

The most prominent observation from the 1st 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 human player and the first AI team (whereas the difference between the first and last human player is merely 6.00 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 reach of AI systems.

B. Modular approach dominates this competition

All three AI teams adopted modular approaches to the 1st 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 two points across all three navigation systems.

1) 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 the 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, 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.

2) 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 cloud, 2D costmap, and BEV map. 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.

C. Learning is an essential component for each team

Despite the lack of end-to-end learning approaches in the 1st ERC, machine learning is still widely used in the 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 only to the perception tasks show the promises 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 is available in the future, why and how learning should be applied to navigation still needs to be carefully considered by the robotics community [17].

D. 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.

E. Challenges of low-cost mobile robots in the wild

One unique feature of 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.

1) 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 IMU or wheel encoder is necessary and requires precise calibration for each deployment. While calibration data is 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 track or the depth or 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 re-calibration techniques that do not require dedicated calibration procedure [21], such as utilizing Structure-from-Motion to dynamically determine camera intrinsic parameters during initialization.

2) Unreliable GPS: As observed in the challenge, GPS quality varies across the globe and is of particularly low quality for the robots located in, e.g., Abu Dhabi. Without RTK fixation, blindly trusting unreliable GPS will significantly jeopardize localization and odometry, causing trouble for planning and control. Interestingly, such a problem has also caused trouble for some human players to determine the 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.

REFERENCES

[1] 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.

[2] 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.

[3] X. Xiao, Z. Xu, Z. Wang, Y. Song, G. Warnell, P. Stone, T. Zhang, S. Ravi, G. Wang, H. Karnan *et al.*, "Autonomous ground navigation in highly constrained spaces: Lessons learned from the benchmark autonomous robot navigation challenge at icra 2022 [competitions]," *IEEE Robotics & Automation Magazine*, vol. 29, no. 4, pp. 148–156, 2022.

[4] X. Xiao, Z. Xu, G. Warnell, P. Stone, F. G. Guinjoan, R. T. Rodrigues, H. Bruyninckx, H. Mandala, G. Christmann, J. L. Blanco-Claraco *et al.*, "Autonomous ground navigation in highly constrained spaces: Lessons learned from the second barn challenge at icra 2023 [competitions]," *IEEE Robotics & Automation Magazine*, vol. 30, no. 4, pp. 91–97, 2023.

[5] X. Xiao, Z. Xu, A. Datar, G. Warnell, P. Stone, J. J. Damanik, J. Jung, C. A. Deresa, T. D. Huy, C. Jinyu *et al.*, "Autonomous ground navigation in highly constrained spaces: Lessons learned from the third barn challenge at icra 2024 [competitions]," *IEEE Robotics & Automation Magazine*, vol. 31, no. 3, pp. 197–204, 2024.

[6] Starship, "Starship," https://www.starship.xyz/, 2024.

[7] kiwibot, "kiwibot," https://www.kiwibot.com/, 2024.

[8] Tiny Mile, "Tiny mile," https://tinymile.ai/, 2024.

[9] FrodoBots, "Frodobots-2k - datasets at hugging face," https://huggingface.co/datasets/frodobots/FrodoBots-2K, 2024.

[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 Transactions on Robotics*, vol. 37, no. 6, p. 1874–1890, Dec. 2021. [Online]. Available: http://dx.doi.org/10.1109/TRO.2021.3075644

[11] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.

[12] M. Xu, Z. Zhang, F. Wei, H. Hu, and X. Bai, "Side adapter network for open-vocabulary semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2945–2954.

[13] M. Hu, W. Yin, C. Zhang, Z. Cai, X. Long, H. Chen, K. Wang, G. Yu, C. Shen, and S. Shen, "Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation," *arXiv preprint arXiv:2404.15506*, 2024.

[14] L. Mao, G. Warnell, P. Stone, and J. Biswas, "Pacer: Preference-conditioned all-terrain costmap generation," 2024. [Online]. Available: https://arxiv.org/abs/2410.23488

[15] S. Nasiriany, F. Xia, W. Yu, T. Xiao, J. Liang, I. Dasgupta, A. Xie, D. Driess, A. Wahid, Z. Xu *et al.*, "Pivot: Iterative visual prompting elicits actionable knowledge for vlms," *arXiv preprint arXiv:2402.07872*, 2024.

[16] A. J. Sathyamoorthy, K. Weerakoon, M. Elnoor, A. Zore, B. Ichter, F. Xia, J. Tan, W. Yu, and D. Manocha, "Convoi: Context-aware navigation using vision language models in outdoor and indoor environments," *arXiv preprint arXiv:2403.15637*, 2024.

[17] 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.

[18] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[19] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE transactions on robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[20] Z. Teed, L. Lipson, and J. Deng, "Deep patch visual odometry," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[21] X. Xiao, Y. Zhang, H. Li, H. Wang, and B. Li, "Camera-imu extrinsic calibration quality monitoring for autonomous ground vehicles," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4614–4621, 2022.