2023 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)
June 28-30, 2023. Seattle, Washington, USA

Real-Time Visual-Servo Navigation for Map-Free Self-Driving in Unstructured Outdoor Environments

Ho-Feng Chang and Chih-Hung G. Li*, *Member, IEEE*

Abstract— **This paper presents a novel navigation system for unstructured outdoor environments that does not rely on pre-existing maps. The system employs a responsive action design that combines a deep Convolutional Neural Network (ConvNet) for evaluating traversable regions based on RGB inputs, GNSS for global coordinates, and a compass. The Global Sense (GloS) module and the Traversable Region Abbreviation ConvNet (TRAC) work in tandem, with the former tracking the destination's relative position and the latter determining the robot's position within the traversable region. The action maker then executes Grand Direction and Local Maneuver simultaneously until the destination is reached. The system also uses deep learning-based semantic segmentation to analyze front-view images, which are then passed to the lightweight TRAC for real-time execution on an embedded system. Our experiments show that TRAC achieved an accuracy of over 70% at a frame rate of 30 fps. We have implemented the proposed system on a mobile robot and conducted field tests on a university campus, demonstrating the feasibility of map-free navigation with the proposed system.**

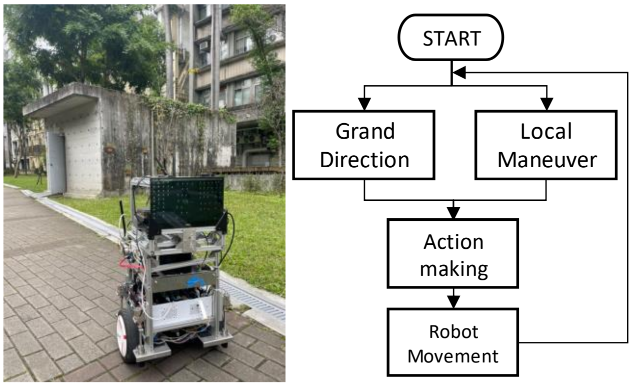


Fig. 1. To conduct map-free autonomous navigation in unstructured outdoor environments, the proposed system employs GNSS and compass to track the destination's location, while the Traversable Region Abbreviation ConvNet provides the real-time field information for movement controls.

I. Introduction

The rising demand for B2C logistics has spurred interest in autonomous mobile systems, such as ground and aerial vehicles, for transportation. Autonomous ground vehicles must navigate through various environments, including indoor spaces and outdoor areas like city streets, highways, and unstructured environments like campuses, parks, and factory compounds. While city streets and highways provide clear guidance through the use of lane lines and traffic signs, unstructured environments pose a greater challenge due to the diverse conditions and lack of clear boundaries. Navigation in these environments typically requires the use of Simultaneous Localization and Mapping (SLAM) techniques for map generation, as detailed path planning through readily available maps is often not feasible. The need for quick delivery to previously unvisited territories further complicates navigation in unstructured environments.

In this paper, we propose a general-purpose architecture for autonomous mobile robots (AMRs) to navigate unstructured environments without pre-constructed maps. Our architecture uses GNSS (Global Navigation Satellite System) coordinates and a compass for position and direction, respectively. We developed a visual-servo system using an onboard camera and a Convolutional Neural Network (ConvNet)-based classifier to detect traversable regions ($\mathcal{R}$) in real-time. The two-level action-making architecture guides the robot's movement, with Grand Direction tracking the destination direction and Local Maneuver guiding the robot within the traversable region, as depicted in Fig. 1. Our contributions include: a map-free navigation system for guiding AMRs in unstructured outdoor environments using RGB vision and global positioning, an automated method for training the ConvNet-based front-view inspector through semantic segmentation, and experimental data from field trials demonstrating the effectiveness of the system.

II. Related Work

Computer vision technologies for end-to-end vehicle steering have been extensively researched, with early systems like ALVINN [1] and MANIAC using neural networks for road-following steering. LeCun et al. [2] demonstrated the feasibility of end-to-end steering for off-road obstacle avoidance using a ConvNet, followed by numerous deep learning methods developed for autonomous driving, including for indoor [3-6] and outdoor [7, 8] navigation and exploration in unstructured environments [9]. Some addressed

*This work was supported by the Ministry of Science and Technology of the Republic of China, Taiwan, under Contract No.: MOST 111-2221-E-027-106-MY2.

Ho-Feng Chang is with the Graduate Institute of Mechatronic Engineering, National Taipei University of Technology, Taipei, 10608 Taiwan ROC (phone: +886-2-2771-2171 ext. 2080; fax: +886-2-2776-4889; e-mail: win22127608@gmail.com).

Chih-Hung G. Li is with the Graduate Institute of Manufacturing Technology, National Taipei University of Technology, Taipei, 10608 Taiwan ROC (phone: +886-2-2771-2171 ext. 2092; fax: +886-2-2776-4889; e-mail: cL4e@mail.ntut.edu.tw).

978-1-6654-7633-1/23/$31.00 ©2023 IEEE

the importance of social compliance [5, 10]. The mediated perception [8] and behavior reflex [2, 3] approaches have been widely used, with direct perception proposed as a lightweight alternative by Chen et al. [8]. In this study, we propose a framework that inputs RGB images of the robot's front view to a deep ConvNet for real-time navigation in unstructured environments.

Navigating in unstructured environments presents distinct challenges compared to street navigation, such as the diversity of visual features that cannot be condensed into a few representative landmarks. Additionally, while street maps are commonly available, map generation for unstructured environments requires additional effort. For example, Provodin et al. [7] proposed an image segmentation method that uses a stereo system to distinguish the drivable region in off-road environments. Other approaches include 3D mapping to identify traversable regions using wheel odometry, 2D laser, and RGB-D data [11], and fusing LiDAR measurements and RGB perceptions in a dual-input ConvNet [8]. SLAM has also been widely used for unstructured environments [12], where sensors perceive the environment and build a map to simultaneously estimate the robot's position.

Accurate geometric mapping can be beneficial for path planning and localization, but it often comes at the cost of large data storage and computational resources. In contrast, the map-free approach is particularly attractive for exploring areas where detailed maps and the time to generate them are unavailable. Wireless sensor networks have been used for map-free navigation [13], but deploying such networks can be time-consuming. To address the challenges of cluttered indoor environments, a map-free navigation system using a depth camera, deep neural networks, and reinforcement learning was proposed for a quadrotor [14]. The approach of Muñoz-Bañón et al. [15] divided autonomous navigation into node-based global path planning and local path planning, where a topological map reduced the need for a detailed geometric map but still required some preparation. Local path planning deals with navigation between nodes, 3D LiDAR scans were utilized to generate detailed local models, and the Naive-Valley-Path method was proposed to produce a naive cost map representation and infer an optimal path. A part of our approach addresses a similar problem of local path planning, but instead of utilizing depth sensors, we proposed an approach based on semantic segmentation [16] of pure RGB vision. However, semantic segmentation can be difficult to execute in real-time on embedded systems, thus we adopted the direct perception approach to transfer the learned knowledge of semantic segmentation to a more lightweight ConvNet that can run in real-time.

III. METHOD

The goal of this study is to create an automatic navigation system for outdoor unstructured environments that does not require any prior knowledge of the field. The proposed system includes two levels of decision-making: tracking the global direction of the destination and navigating locally within the traversable region, and three sensory components: a GNSS system that provides the robot's coordinates, an onboard compass that indicates the robot's moving direction, and a visual recognition system that detects the condition of the front environment. An integrated action maker receives input from the three sensory components and generates immediate action commands based on the active action policy.

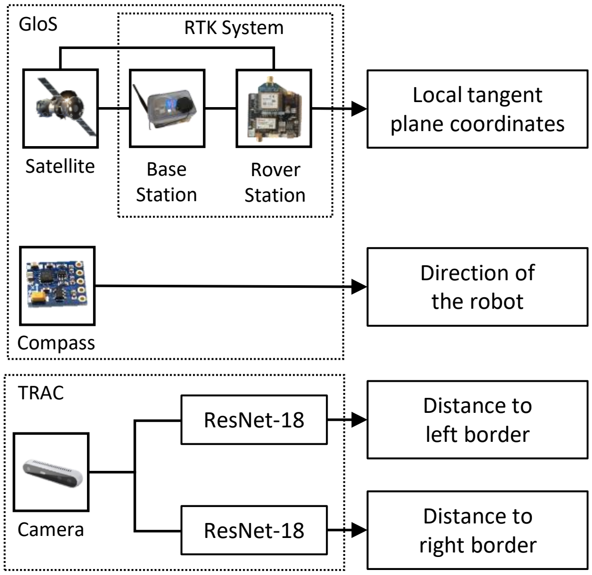


Fig. 2. Illustration of the proposed navigation framework including TRAC and GloS.

A. *Navigation Framework*

Fig. 2 illustrates the navigation framework. Upon initialization, the onboard GNSS module measures the robot's global coordinates. By comparing the coordinates of the destination and the robot, the spatial vector from the robot to the destination is obtained. The angle between the destination vector and the robot's heading direction is then calculated, and the robot is rotated to face the destination as its initial pose.

The robot then navigates to the destination by exploring the traversable region using the traversable region abbreviation ConvNet (TRAC) and the global sense (GloS) module jointly. At any instance i, the robot is located at x_i, TRAC generates key geometric measurements ω_i of the traversable region in front of the robot, while GloS continuously updates the relative position of the destination σ_i. Based on this information, the action maker generates suitable actions according to the action policy π,

$$\pi(u_i|x_i) = \tilde{\pi}(u_i|\omega_i,\sigma_i)\Phi(\omega_i|I_i)\Gamma(\sigma_i|x_i) \qquad (1)$$

where Φ denotes TRAC, and Γ denotes GloS.

TRAC is a ConvNet-based visual detector trained to observe the RGB environment in front of the robot I_i and return ω_i, such as the distances from the left or right boundaries of $\mathcal{R}$ to the robot. As will be clearer in III-*C*, the action maker issues motor commands to adjust the robot's relative position in $\mathcal{R}$, with a goal of approaching the destination without leaving the traversable region. The robot's trajectory of N instances is represented by the following equation:

$$\pi(\tau) = p(\boldsymbol{x}_1)\prod_{i=1}^{N} p(\boldsymbol{x}_{i+1}|\boldsymbol{x}_i,u_i) \int \pi(u_i|\boldsymbol{x}_i)\, d\boldsymbol{x}_i \quad (2)$$

At any given time t, Φ assigns a confidence score to each possible visual class v_n for the input vector $\mathbf{v_t}$; the class with the highest score will be the predicted class,

$$\hat{v}_t = argmax_n\big(\Phi(\mathbf{v_t}, v_n)\big). \quad (3)$$

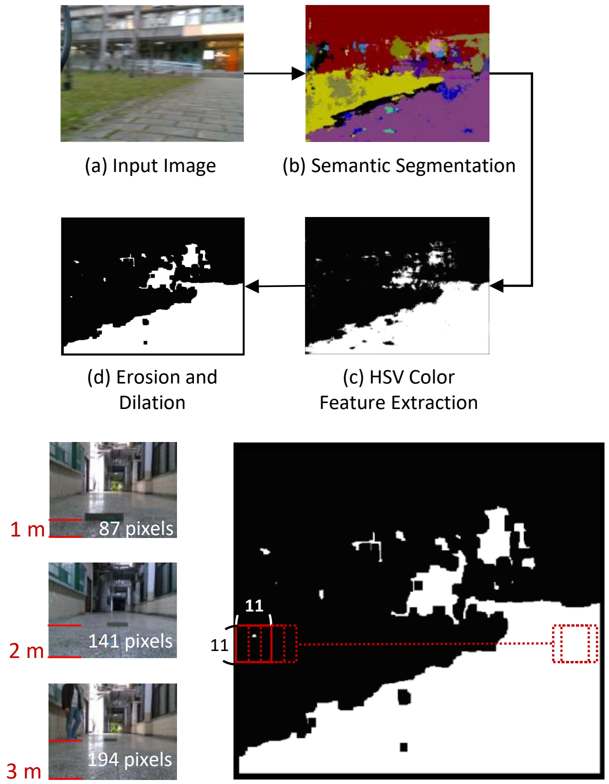
To achieve accurate results, the aim is to have the predicted class as close as possible to the ground-truth $v_{t/g}$,

$$v_t^{\,*} = argmin_{\hat{v}_t}\left\|\hat{v}_t - v_{t/g}\right\|_2. \quad (4)$$

GloS integrates the GNSS module and the compass to accurately predict the robot's pose relative to the destination. We adopted the RTK (Real-Time Kinematic positioning) system [17] to enhance the precision of global positioning. Our experiments show that with a base station placed within a radius of 1 km, the positioning of the rover station can be obtained with a precision of 6 m.

B. *Traversable Region Abbreviation ConvNet*

Due to the large number of visual varieties in outdoor unstructured environments, deep learning-based semantic segmentation models trained with sufficient examples appear to be a promising solution [16, 18]. However, semantic segmentation still requires high computational resources and can be difficult to perform in real time on embedded systems. Additionally, the approach often involves a high demand for manual labor in collecting and annotating training data. To address these issues, we developed TRAC to transfer the results of semantic segmentation to a much lighter ConvNet that satisfies the requirements for high frame rate and low computational resources.



(a) Input Image

(b) Semantic Segmentation

(d) Erosion and Dilation

(c) HSV Color Feature Extraction

(e) Using an 11 × 11 kernel to scan the image at the 2 m depth.

Fig. 3. The training process of TRAC involving semantic segmentation and key parameter annotation. In this example, the label for left TRAC is 0; the label for right TRAC is 4.

Specifically, we adopted a semantic segmentation module to analyze the collected environment images in order to separate the traversable regions at the pixel level. We then extracted key metrics from the resulting $\mathcal{R}$, such as the distances from the image center to the left and right borders of $\mathcal{R}$. By annotating the original environment images with these metrics, we trained a visual detector using ResNet-18 as the backbone.

The data preparation process for training is shown in Fig. 3. We collected a total of 160,000 images, which included various scenarios on a university campus. The images were divided into 32 classes using a semantic segmentation module [19]. The road and sidewalk classes were designated as traversable and assigned the color white, while the rest of the image was assigned the color black. We performed erosion and dilation operations to remove any image noise. Using the camera's viewing angles and vertical position as reference, we constructed a 3D model to measure the distance from the center of the image to the left and right boundaries of the road. As illustrated in Fig. 3, the features one meter in front of the camera are represented by the 87th pixel from the bottom of a 480-pixel image. We used an 11x11-pixel kernel to examine the image at a constant height and obtain the distance of the detected boundary to the center of the image. The road boundaries were captured two meters ahead of the robot for TRAC judgment. The left and right sides were detected separately and divided into five categories based on the distance D. The division was not proportional, as described in (5), with increased vigilance near the boundaries.

$$
\begin{aligned}
&\text{Label 0:} && \mathcal{D} < 107 \text{ mm}\\
&\text{Label 1:} && 107 \text{ mm} \le \mathcal{D} \le 268 \text{ mm}\\
&\text{Label 2:} && 268 \text{ mm} \le \mathcal{D} \le 488 \text{ mm}\\
&\text{Label 3:} && 488 \text{ mm} \le \mathcal{D} \le 763 \text{ mm}\\
&\text{Label 4:} && 763 \text{ mm} < \mathcal{D} \qquad (5)
\end{aligned}
$$

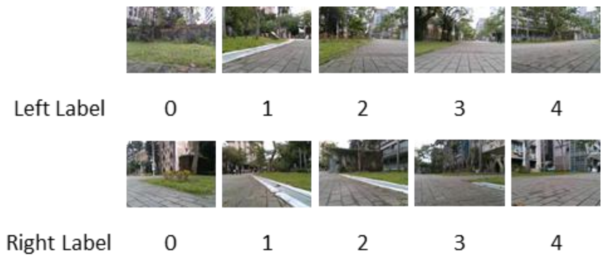


Fig. 4. These images are samples used to train the left and right TRAC models, which were captured by the single camera mounted on the robot.

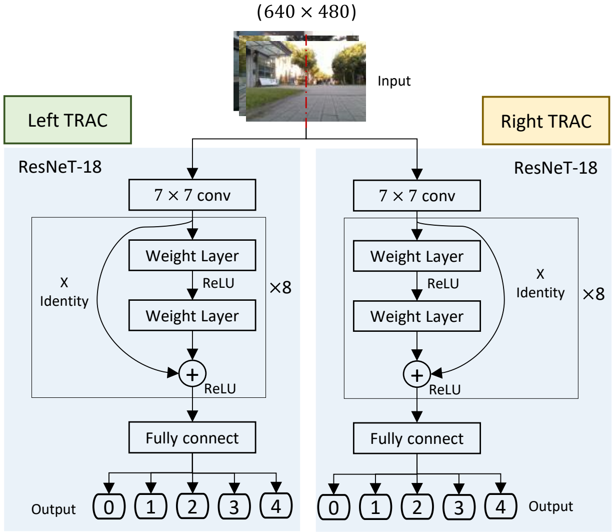


Fig. 5. Illustration of the architecture of TRAC, which utilizes ResNet-18 and outputs five classes. TRAC runs in parallel with two separate networks, one for detecting the right border and one for the left border.

Fig. 4 presents examples of the training images used for TRAC. Two sets of data, each containing five categories, were assembled to indicate the left and right boundary information separately. Fig. 5 illustrates the ConvNet of TRAC, which adopts ResNet-18 [20] as its backbone. TRAC uses two separate networks, one for left boundary prediction and the other for right boundary prediction. As TRAC detects the robot's position relative to the borders of $\mathcal{R}$, such information allows the subsequent action maker to determine steering actions for maneuvering the robot along ideal paths. Labels 0 and 4 indicate the two extreme conditions of being very close to the border and being far from the border. Label 0 typically signals the need for steering away from the border, while Label 4 is used to indicate the possible presence of a road intersection. In this way, TRAC not only provides measurements to facilitate in-path navigation, but also informs the higher-level path planning about available options for direction correction towards the destination.

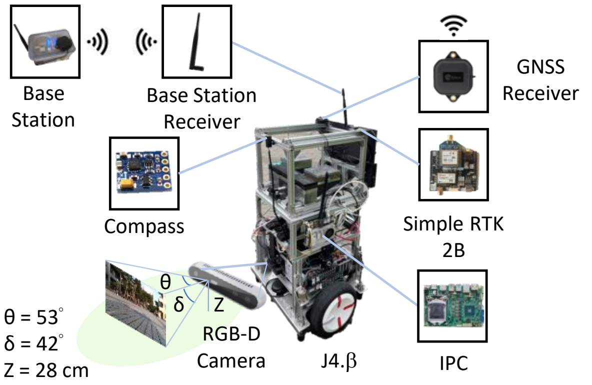
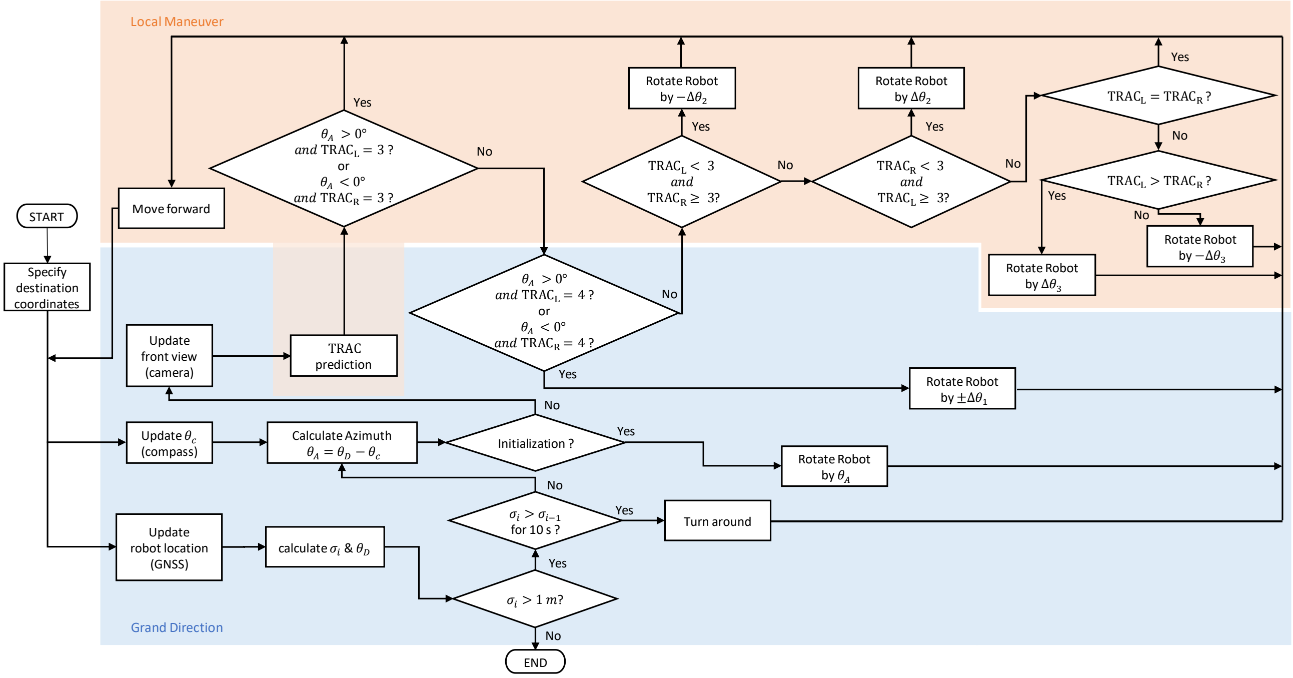


Fig. 6. Illustration of the design of the mobile robot, J4.β, utilized for the field tests.

C. Decision-Making and Robot Control

We implemented the proposed system on a mobile robot J4.β [21], which is a two-wheeled self-balanced mobile robot with its direction controlled by differential speed and its movement by altering the position of the center of gravity [5]. The self-balanced mobile platform allows it to traverse both indoor and outdoor environments [22]. An RGB-D camera –



Local Maneuver
START
Specify destination coordinates
Move forward
Yes
$\theta_A > 0°$ and $\mathrm{TRAC_L} = 3$? or $\theta_A < 0°$ and $\mathrm{TRAC_R} = 3$?
No
Rotate Robot by $-\Delta\theta_2$
Yes
$\mathrm{TRAC_L} < 3$ and $\mathrm{TRAC_R} \geq 3$?
No
Rotate Robot by $\Delta\theta_2$
Yes
$\mathrm{TRAC_R} < 3$ and $\mathrm{TRAC_L} \geq 3$?
No
Yes
$\mathrm{TRAC_L} = \mathrm{TRAC_R}$?
No
$\mathrm{TRAC_L} > \mathrm{TRAC_R}$?
Yes
No
Rotate Robot by $-\Delta\theta_3$
Rotate Robot by $\Delta\theta_3$
$\theta_A > 0°$ and $\mathrm{TRAC_L} = 4$? or $\theta_A < 0°$ and $\mathrm{TRAC_R} = 4$?
No
Yes
Update front view (camera)
TRAC prediction
Rotate Robot by $\pm\Delta\theta_1$
No
Update θ_c (compass)
Calculate Azimuth $\theta_A = \theta_D - \theta_c$
Initialization ?
Yes
Rotate Robot by θ_A
No
$\sigma_i > \sigma_{i-1}$ for 10 s ?
Yes
Turn around
Update robot location (GNSS)
calculate σ_i & θ_D
Yes
$\sigma_i > 1\ m$?
No
END
Grand Direction

Fig. 7. Block diagram of the action-making procedure, in which Grand Direction directs the robot toward the target, and Local Maneuver guides the robot to navigate within $\mathcal{R}$.

Intel RealSense D415 was adopted as the visual device (see Fig. 6). Although the camera has depth detection capability, it was not used here.

Autonomous navigation involves executing action commands through the use of a block diagram, as shown in Fig. 7. The navigation process involves two parallel levels of decision-making: the Grand Direction level and the Local Maneuver level. The Grand Direction level uses GloS to determine the real-time relative position of the destination and to adjust the robot's path as needed. The Local Maneuver level guides the robot to move within $\mathcal{R}$ and uses the outputs from TRAC to ensure a safe distance from the boundaries of $\mathcal{R}$. The robot typically travels along the boundary of $\mathcal{R}$ near the destination until a new path is required by the Grand Direction level.

The proposed map-free navigation is depicted in Fig. 8. The first step is for GloS to identify the destination's direction and rotate the robot to face toward the destination. The robot then moves towards the destination until it encounters the boundary of $\mathcal{R}$. Along the way, GNSS provides the distance σ_i and direction θ_D from the current position to the destination, while the compass provides the robot's direction θ_c. Based on the TRAC results, different levels of direction fine-tuning ($\Delta\theta_1$,$\Delta\theta_2$,$\Delta\theta_3$) are issued. For example, a more delicate maneuver is necessary near the road edge, and thus $\Delta\theta_2$ is smaller than $\Delta\theta_3$. In the left scenario shown in Fig. 8, the robot encounters the left boundary while in the right scenario, it encounters the right boundary. In these situations, the Local Maneuver level directs the robot to stay within the boundary of $\mathcal{R}$ by adjusting its steering along the borders. However, this may cause the robot to deviate from its intended destination. To address this issue, the Grand Direction level continuously evaluates θ_A and issues a turning command when TRAC indicates a more suitable path. In the left scenario of Fig. 8, the robot turns left, while in the right scenario, it turns right. By coordinating the actions of the Grand Direction and Local Maneuver levels, the robot can eventually reach its desired destination.

IV. Experiments

We conducted two sets of experiments: one to evaluate TRAC, and the other to test the actual navigation in the field. TRAC was trained on a server computer equipped with an AMD Ryzen 9 3900X 12-Core Processor, 64G RAM, and an NVIDIA GeForce RTX 2080Ti GPU. During the training, 1% of the images were set aside for validation. After 150 training epochs, the training accuracy reached 0.98, and the validation accuracy was around 0.88. The total training time was approximately 180 minutes.

A. TRAC Accuracy

To assess the performance of TRAC, we gathered 20,000 new images of different environments not used for training purposes. We then applied the same segmentation procedure to these images to obtain ground-truth labels. TRAC was tested on these images, and the outcomes are presented in Fig. 9. The Left TRAC produced an average recall, precision, and accuracy of 0.62, 0.63, and 0.62, respectively. The Right TRAC generated 0.61, 0.61, and 0.60. The detections of labels 0, 3, and 4 showed higher accuracy, which are also more critical for successful operations. For instance, label 0 indicates an approaching boundary, and label 4 denotes the availability of a junction. In both the left and right models, these three labels had an accuracy of more than 70%. Errors in semantic segmentation that caused mislabeled training images might be the reason for the less satisfactory outcomes on labels 1 and 2.

B. Field Tests

We conducted field tests at a university campus, as shown in Fig. 10, to evaluate J4.β's ability to navigate between various structural features, such as grass, wood decks, stairs, untraversable brick ground, and sculptures. We selected five locations on the map and tested each route ten times, counting a trial as successful if the robot remained inside the traversable region without human intervention. All computations were performed on the onboard PC, using an Intel 9th gen i7 CPU and 16 GB DDR4. We converted the TensorFlow model of TRAC to an IR model using the Intel OpenVINO Toolkits, obtaining an average inference speed of 30 fps.

Table I summarizes the field test results, indicating an overall success rate of approximately 65%. Some routes had success rates higher than 90%, while the worst was below 50%. Detection errors on stairs and wood decks were the primary cause of failure for the routes between A and C. J4.β performed well in detecting grass and plantings, resulting in a higher success rate for routes of B to D and C to B. Enlarging the training database can enhance TRAC's performance in the future. Another significant factor that affected the success rate was the existence of small lanes that led the robot to dead-ends, resulting in unresolvable circling. For example, when

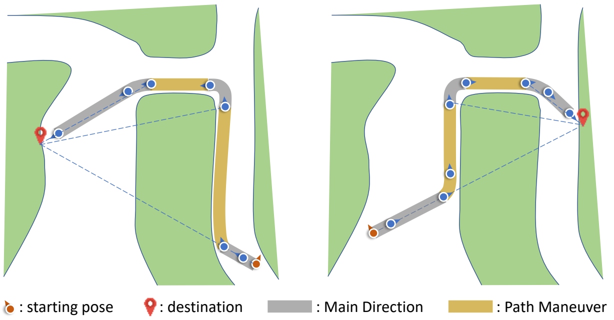
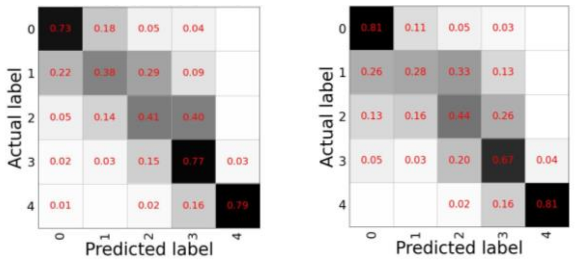


Fig. 8. Illustrations of the navigation process with the coordination of Grand Direction and Local Maneuver levels.



(a) Left TRAC confusion matrix

(b) Right TRAC confusion matrix

Fig. 9. The performance of TRAC.

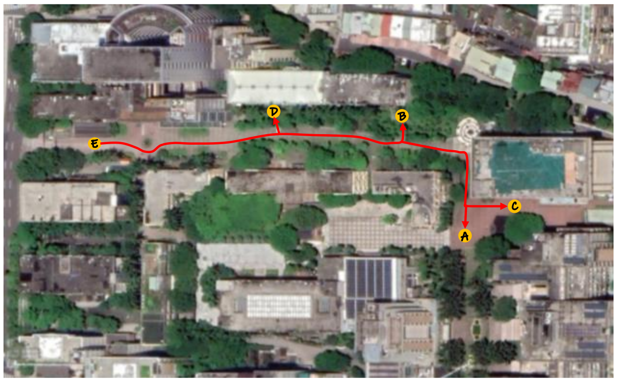


Fig. 10. Illustration of the testing field and the test routes.

TABLE I. RESULT SUMMARY OF THE FIELD TESTS

Section	Success rate	Distance (m)	Features
A → B	70%	150	grass, plantings, road sign
B → A	70%	150	grass, plantings, road sign
A → C	40%	84	stone barrier, wood deck, buildings
C → A	50%	84	stone barrier, wood deck, buildings
B → C	70%	145	grass, plantings, road sign, wood deck, stairs, buildings
C → B	90%	145	grass, plantings, road sign, wood deck, stairs, buildings
B → D	100%	88	grass, plantings
E → B	30%	198	grass, plantings, road sign
A → E	60%	308	grass, plantings, road sign
C → E	70%	303	grass, plantings, road sign, wood deck, stairs, buildings
B → E	50%	198	grass, plantings, road sign

navigating from E to B, the robot sometimes turned left into a dead-end and was unable to resolve the situation. A possible solution is to improve TRAC and GloS to recognize dead-ends or automatically move out of them after a few circling attempts.

V. CONCLUSION

In summary, this article proposes a novel navigation system for unstructured outdoor environments that eliminates the need for pre-existing maps. The system utilizes GNSS, compass, and camera for sensory input and includes a two-level action-making architecture. TRAC, a real-time visual inspection tool for the traversable region, is introduced, achieving an accuracy of over 70% at 30 fps. Field tests achieved an overall success rate of 65%, with misclassification of stairs and wood decks causing critical failures. The study suggests future research to enlarge the TRAC training database and equip the system with dead-end escape capabilities. The video of the field tests can be accessed through this link: https://www.youtube.com/watch?v=LbKAEqbKTXA.

REFERENCES

[1] D. A. Pomerleau, "Knowledge-based training of artificial neural networks for autonomous robot driving," in J. Connell and S. Mahadevan, editors, *Robot Learning*. Kluwer Academic Publishing, 1993.

[2] U. Muller, J. Ben, E. Cosatto, B. Flepp, Y. LeCun, "Off-road obstacle avoidance through end-to-end learning," in *Proc. Adv. Neural Inf. Proc. Syst. 18 (NeurIPS)*, 2005, pp. 739–746.

[3] L. Tai, S. Li, M. Liu, "A Deep-network solution towards model-less obstacle avoidance," in *Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS)*, 2016, pp. 2759–2764.

[4] C. G. Li and L.-P. Zhou, "Training end-to-end steering of a self-balancing mobile robot based on RGB-D image and deep ConvNet," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM 2020)*, 2020/7/6-10, Boston, USA, pp. 898-903.

[5] C. G. Li, L.-P. Zhou, and Y.-H. Chao, "Self-balancing two-wheeled robot featuring intelligent end-to-end deep visual-steering," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 5, pp. 2263-2273, 2021.

[6] N. Patel, A. Choromanska, P. Krishnamurthy, and F. Khorrami, "Sensor modality fusion with CNNs for UGV autonomous driving in indoor environments," in *Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS)*, 2017, pp. 1531-1536.

[7] A. Provodin, L. Torabi, B. Flepp, Y. LeCun, M. Sergio, L. D. Jackel, U. Muller, J. Zbontar, "Fast incremental learning for off-road robot navigation," arXiv:1606.08057 [cs.RO], 2016.

[8] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "DeepDriving: learning affordance for direct perception in autonomous driving," in *Proc. Int. Conf. Comput. Vision (ICCV)*, 2015, pp. 2722–2730.

[9] L. Tai, J. Zhang, M. Liu, and W. Burgard, "Socially compliant navigation through raw depth inputs with generative adversarial imitation learning," in *Proc. IEEE Int. Conf. Robotics Auto. (ICRA)*, 2018, pp. 1111–1117.

[10] L. Tai and M. Liu, "Mobile robots exploration through cnn-based reinforcement learning," *Robot. Biomim*, vol. 3, 24, 2016.

[11] C. Wang et al., "Autonomous mobile robot navigation in uneven and unstructured indoor environments," in *Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS)*, 2017, pp. 109-116.

[12] A. Singandhupe and H. M. La, "A review of SLAM techniques and security in autonomous driving," in *Proc. IEEE Int. Conf. Robot. Comput. (IRC)*, 2019, pp. 602-607.

[13] W. Chen, T. Mei, H. Liang, Z. You, S. Li, and M. Q.-H. Meng, "Environment-map-free robot navigation based on wireless sensor networks," in *Proc. Int. Conf. Inf. Acquisition*, 2007, pp. 569-573.

[14] P. Nitschke, "Reinforcement learning for fast, map-free navigation in cluttered environments using aerial robots," master's thesis, Norges teknisk-naturvitenskapelige universitet, 2022.

[15] M. Á. Muñoz-Bañón, E. Velasco-Sánchez, F. A. Candelas, and F. Torres, "OpenStreetMap-based autonomous navigation with LiDAR Naive-Valley-Path obstacle avoidance," *IEEE Trans. Intell. Trans. Syst.*, vol. 23, no. 12, pp. 24428-24438, 202

[16] K. He, G. Gkioxari, P. Dollar, R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vision (ICCV)*, 2017, pp. 2961-2969.

[17] J. Wang, "Stochastic modeling for Real-Time Kinematic GPS/GLONASS positioning." *Navigation*, vol. 46, no. 4, pp. 297-305, 1999.

[18] S. Kobayashi, Y. Sasaki, A. Yorozu, A. Ohya, "Probabilistic semantic occupancy grid mapping considering the uncertainty of semantic segmentation with IPM," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM 2022)*, 2022, pp. 250-255.

[19] G. Seif, "Semantic segmentation with deep learning," [Online]. Available: https://towardsdatascience.com/semantic-segmentation-with-deep-learning-a-guide-and-code-e52fc8958823, last visited Jan. 25, 2023.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2016, pp. 770-778.

[21] Y.-C. Hsu, M.-C. Lin, and C. G. Li, "Mobility improvement on the two-wheeled dynamically balanced robot – J4.β," in *Proc. IEEE Int. Conf. Auto. Sci. Eng., (CASE)*, 2021, pp. 442–447.

[22] H.-Z. Lin, H.-H. Chen, K. Choophutthakan, and C. G. Li, "Autonomous mobile robot as a cyber-physical system featuring networked deep learning and control," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM 2022)*, 2022, pp. 268-274.