Sensors and Materials, Vol. 37, No. 12 (2025) 5517–5534
MYU Tokyo

S & M 4260

Mobile Robot Based on Visual Semantic Simultaneous Localization and Mapping and Autonomous Navigation System

Pi-Yun Chen, Shao-Ting Yang, Neng-Sheng Pai,* and Yu-Cheng Cheng

Department of Electrical Engineering, National Chin-Yi University of Technology,
No. 57, Sec. 2, Zhongshan Rd, Taiping Dist, Taichung City 41170, Taiwan (ROC)

(Received May 27, 2025; accepted December 3, 2025)

Keywords: deep learning, end-to-end autonomous navigation, semantic segmentation, semantic SLAM, autonomous navigation neural network

In this paper, we present an end-to-end vision-based autonomous navigation system for robots using deep learning. The system integrates semantic segmentation with visual simultaneous localization and mapping (SLAM) to implement a semantic SLAM approach. The generated map not only contains geometric information but also recognizes and classifies objects in the environment, enhancing the robot's perception capabilities. For semantic SLAM, a semantic segmentation neural network is first designed and trained using the SUNRGB-D dataset. Real-time data processing and node registration are carried out using the robot operating system. Depth images and camera intrinsics are used to generate point clouds, and semantic segmentation images are fused into the generated point clouds to create a 3D semantic map using Octomap, enriched with semantic information. For autonomous navigation, a test environment is set up in the Gazebo simulation, and expert data is collected. The autonomous navigation neural network is trained based on color images, depth images, and the robot's position and orientation data, enabling the network to output linear and angular velocities based on visual data alone. Without relying on additional sensors, the robot is capable of path planning, obstacle avoidance, and autonomous navigation purely through visual input.

1. Introduction

With the advancement of Industry 4.0, the surge in AI and its rapid technological progress, particularly breakthroughs in hardware such as Simultaneous Localization and Mapping's advanced process chips, have driven the application and innovation of intelligent automation and autonomous mobility technologies. In recent years, the COVID-19 pandemic and the U.S.–China trade war have led to labor shortages and rising labor costs in the global supply chain and economy. As a result, intelligent automated production systems have become a popular development focus. Among these systems, automated guided vehicles (AGVs) have been the most widely used. However, AGVs primarily rely on ground-based guidance facilities such as

*Corresponding author: e-mail: pai@ncut.edu.tw
https://doi.org/10.18494/SAM5755

ISSN 0914-4935

magnetic tracks, wires, or markers to determine their path. While this guidance method ensures that AGV can operate stably along a preset route, it also limits their operational range.

Compared with traditional AGV, autonomous mobile robots (AMRs) utilize data from onboard sensors such as light detection and ranging (LiDAR) and depth cameras to perform simultaneous localization and mapping (SLAM),[1] constructing an environmental map. With the integration of path planning algorithms, AMRs achieve autonomous movement, which not only reduces costs but also allows for quick changes in task execution.

With advancements in deep learning and reinforcement learning technologies, end-to-end autonomous driving is increasingly approaching the decision-making patterns of human drivers, enabling vehicles to respond appropriately to various road conditions and situations. In recent years, deep learning (DL)[2] technology, particularly convolutional neural network (CNN),[3] has made breakthrough advancements in image recognition and speech processing. These developments have made the implementation of end-to-end autonomous driving technology more feasible and effective. Not only do these technologies enhance the resolution and predictive capabilities of machine learning models, but they also improve the adaptability and autonomous navigation abilities of vehicles in unfamiliar environments. By leveraging these machine learning algorithms, end-to-end systems can more accurately predict changes in road conditions and adjust their driving strategies in real time to ensure safe and effective driving.

2. Related work

The literature review in this study is divided into four sections, which are briefly summarized as follows: sensors, semantic segmentation, SLAM, and end-to-end autonomous driving navigation.

2.1 Sensors

In the research and development of autonomous driving and AMRs, traditional systems often use various sensors such as radar LiDAR, ultrasonic sensors, and cameras to acquire environmental data. While this approach provides a comprehensive perception of the surrounding environment, it also increases the complexity and cost of the system. With the advancement of hardware technology and deep learning, vision-only navigation systems are becoming increasingly popular in the fields of autonomous driving and AMRs. For example, Tesla utilizes camera images processed through deep learning methods to achieve autonomous driving functionality in vehicles.[4]

2.2 Semantic segmentation

Semantic Segmentation[5] refers to the technique of classifying every pixel in an image, commonly used in fields such as path recognition in autonomous driving and tumor detection in medical imaging. In convolutional neural networks, the last layer typically uses fully connected

layers to output classification probabilities, which cannot predict the classification of every pixel, making them unsuitable for semantic segmentation. Fully convolutional networks (FCN) [5] are considered the pioneers of semantic segmentation, converting traditional fully connected layers into convolutional layers to perform pixel-level classification, laying the foundation for subsequent semantic segmentation neural networks. U-Net,[6] with its distinctive symmetrical structure and successful application in medical image segmentation, has garnered significant attention. PSPNet[7] introduced the pyramid pooling module, effectively aggregating contextual information from different regions, enhancing the model's ability to understand scenes. SENets[8] significantly improve the network's feature representation capability by introducing the channel attention mechanism. ERFNet[9] enhances efficiency and accuracy in real-time applications with its residual factorization design. ESANet[10] combines spatial attention mechanisms to optimize the extraction of key information from large volumes of visual data, excelling particularly in complex visual scenes. The DeepLab[11] series expands the receptive field through dilation convolution, improving the recognition of complex scenes without increasing the number of parameters. Notably, DeepLab v3 Plus[12] introduces an improved atrous spatial pyramid pooling (ASPP) and encoder-decoder module, enhancing the model's ability to recognize objects at multiple scales.

2.3 SLAM

SLAM technology is mainly divided into two categories: Laser SLAM and Visual SLAM. Although both types of SLAM aim to address the localization and mapping challenges in autonomous navigation systems, they differ in technical implementation and application fields. Laser SLAM acquires data by measuring the distance between laser beams and surrounding objects. For example, Grisetti *et al.* and others proposed Gmapping,[13] which uses a 2D-Lidar sensor to build a two-dimensional map based on the Rao-Blackwellized particle filter (RBPF) approach. However, this method relies on odometry data, making it unsuitable for robots operating in uneven terrain. Kohlbrecher *et al.* proposed Hector SLAM,[14] which uses the Gauss-Newton method for feature matching. This algorithm does not require odometry data, allowing robots to function in uneven terrain as well. Visual SLAM, on the other hand, captures visual data from the environment using cameras and utilizes image processing techniques to recognize feature points for localization and mapping. For instance, Mur-Artal *et al.* introduced ORB-SLAM,[15] which builds a 3D map using Oriented FAST and Rotated BRIEF (ORB) feature points based on the PTAM framework. It optimizes keyframe and map construction methods, enhances map initialization and loop closure detection, and prevents cumulative errors caused by movement, achieving excellent mapping results. Labbé and Michaud proposed RTAB-Map,[16] a loop closure detection SLAM with memory management, capable of long-term map updates without losing mapping details, meeting the requirements for long-term and large-scale environment construction. By integrating advanced techniques such as deep learning, Visual SLAM[17–19] can achieve more accurate 3D environment map construction.

2.4 End-to-end autonomous driving navigation

End-to-end autonomous driving navigation systems streamline the process by using neural networks to learn directly from sensor data to control commands, enhancing the system's ability to adapt to unknown environments. This concept can be traced back to 1989, when Pomerleau developed ALVINN,[20] which used images from a camera and laser rangefinder as inputs and employed a neural network to control vehicle movements. This demonstrated that a neural network trained through end-to-end learning can be applied to a vehicle. NVIDIA's PilotNet[21] expanded this idea by learning human driving decisions from visual inputs captured by a front-facing camera, enabling the neural network to derive control commands for the vehicle from visual data. Tesla's full-self driving (FSD) technology, particularly in version 12,[22] represents the latest evolution of this approach. It moves away from traditional rule-based systems for recognizing roads and pedestrians, instead inputting camera images directly into a neural network that outputs control commands such as acceleration, deceleration, braking, and steering, thus achieving end-to-end autonomous driving.

3. Methodology

This paper is divided into two main parts to complete all tasks using pure vision. The first part combines the characteristics of semantic segmentation neural networks for image classification with visual SLAM to implement semantic SLAM. The second part utilizes deep learning to achieve an end-to-end autonomous navigation system. As shown in Fig. 1, the system architecture involves the robot collecting data from depth cameras and odometry to realize environmental perception and map building for the SLAM system. Subsequently, the constructed map and depth camera data are transmitted to the computer via the ROS system for path planning. Finally, a deep learning model is run to complete autonomous navigation.

3.1 Visual semantic segmentation SLAM

In this section, we will explore how to integrate semantic segmentation with feature-based visual SLAM for map building and localization. Traditional SLAM methods produce maps that only contain geometric shapes without semantic distinctions, which limits the robot's

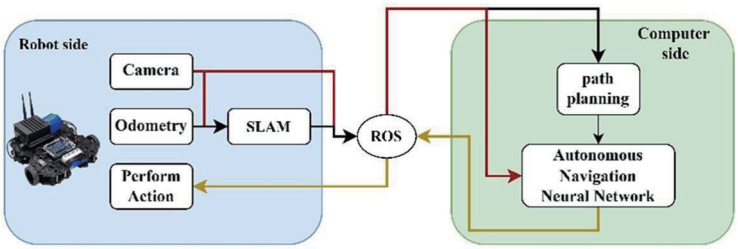


Fig. 1. (Color online) System Architecture Diagram.

understanding of its surrounding environment. With the rapid development of deep learning technologies, neural networks have achieved significant performance levels in semantic segmentation. By combining semantic segmentation networks with SLAM systems, it is possible to create maps enriched with semantic information, enhancing the robot's perception and understanding of the environment. Semantic segmentation can also serve as visual data for robot navigation.

The system integrates semantic segmentation with feature-based visual SLAM, as illustrated in Fig. 2. The system inputs include color and depth images from the camera, and the output is a 3D semantic map. The system is divided into two parts. The first part is semantic segmentation, where the system inputs color and depth images into a semantic segmentation neural network. The network processes the images to generate semantic segmentation images, annotating the categories of various objects within these images. The second part involves integrating semantic segmentation with visual SLAM. Visual SLAM follows the approach of Chen Shian,[23] where the system simultaneously performs feature extraction on RGB images and tracks the camera's motion trajectory based on feature points. Finally, point clouds are generated from the input depth images and camera intrinsic parameters. The semantic segmentation images are then registered to the generated point clouds. Using the generated point clouds, an Octomap is created to produce a 3D semantic map that integrates semantic information.

3.2 Semantic SLAM

Semantic SLAM integrates semantic data into the geometric mapping of the environment, generating a multidimensional map for understanding the environment. The 3D semantic map is represented by point clouds, which are created from color images in the image coordinate system and depth values in the camera coordinate system. Initially, the world coordinate system (X_w, Y_w, Z_w) is converted to the camera coordinate system (X_c, Y_c, Z_c) to obtain the raw depth image. This is then transformed to the image coordinate system (u, v) to acquire the depth-aligned color depth image.

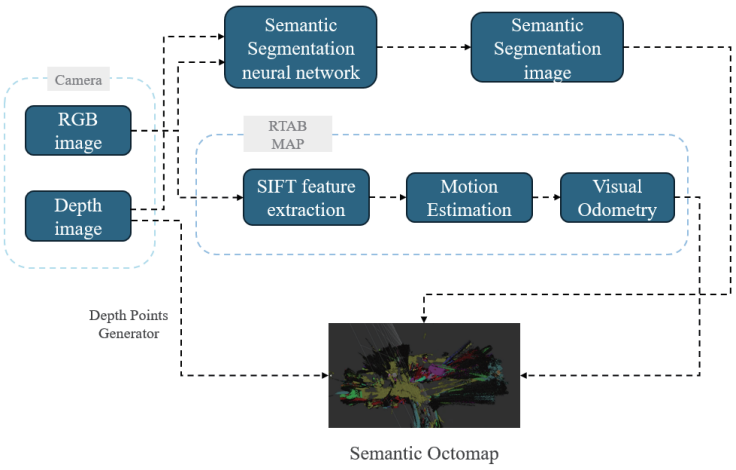


Fig. 2. (Color online) Visual semantic segmentation SLAM system architecture diagram.

$$m_c = [R \mid T] M_w \tag{1}$$

Equation (1) describes the process of transforming from the world coordinate system to the camera coordinate system. m_c represents the point in the camera coordinate system, M_w is the point in the world coordinate system, and $[R \mid T]$ is the transformation matrix from the world coordinate system to the camera coordinate system.

$$m_{uv} = K m_c \tag{2}$$

Equation (2) describes the process of transformation from the camera coordinate system to the image coordinate system. m_{uv} represents the projection location in the image coordinates, and K is the camera intrinsic matrix, which can be obtained from the Realsense D435i camera SDK.

In this paper, point clouds are generated with the camera as the reference point. Pixels in the image coordinates are transformed back into three-dimensional space. On the basis of the projection direction of the unit vector m_c, the depth value of m_{uv} is used to recover the pixel point from two-dimensional to three-dimensional space. The transformation from image coordinates to three-dimensional space is described by

$$M_{fc} = \frac{m_c \times depth\left(m_{uv}\right)}{\left\| m_c \right\|_2}, \tag{3}$$

where M_{fc} represents the position in three-dimensional vector space, $\|m_c\|_2$ is the Euclidean norm of m_c, and $depth(m_{uv})$ is the depth value of the pixel location in the image coordinates. The resulting point cloud in three-dimensional space is illustrated in Fig. 3.

Considering the substantial space required for point cloud mapping, which can reduce overall computational speed, and the presence of many unnecessary features in point clouds, we utilized Octomap[24] for 3D map representation, employing an octree structure to save space. In an octree, each node represents a voxel, with its size determined by the tree structure. Each parent node is subdivided into eight child nodes until the finest resolution is reached. An illustration of the octree structure is shown in Fig. 4.

The Octomap system updates the voxel occupancy probabilities using a logarithmic odds

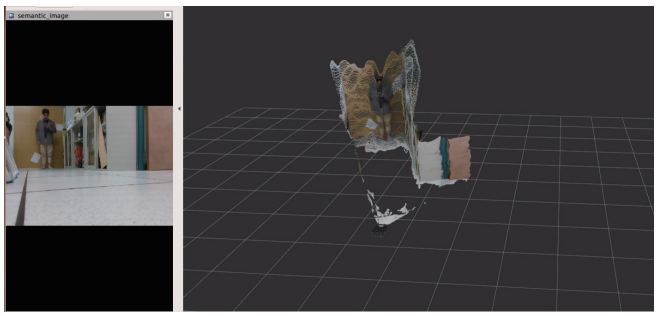


Fig. 3. (Color online) 3D space point cloud.

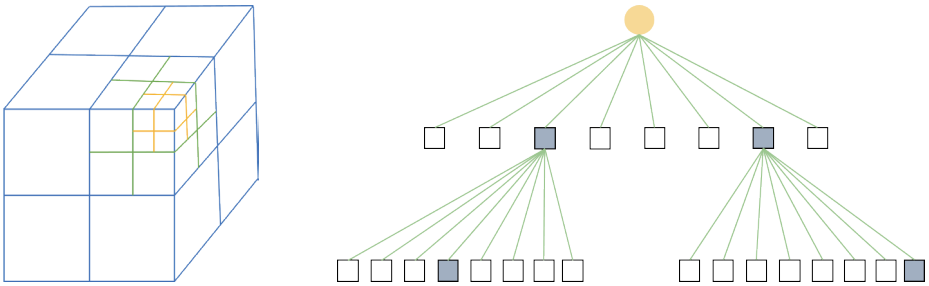


Fig. 4. (Color online) Octree used by octomap.

method. New observations are added to the previous logarithmic odds, and the occupancy probability of a parent node in the octree is computed using the values of its child nodes, typically by methods such as averaging or taking the maximum value. The 3D semantic map is shown in Fig. 5.

3.3 Deep learning navigation and obstacle avoidance

In the real world, conducting experiments with actual robots can be time-consuming and costly. Therefore, training in a simulated environment and then transferring the learned model to the real world can help reduce the time and cost of physical experiments. In this study, we set up a 7 × 7 m environmental map in the simulation world, as shown in Fig. 6, with five immovable obstacles. The goal is to enable the robot to avoid obstacles based on the actual camera perception data during autonomous movement. Additionally, ROS is used to integrate the required sensor equipment in the simulated environment.

3.3.1 Data collection

In this paper, we aim to explore a pure vision-based end-to-end autonomous navigation system, using supervised learning to train neural networks. The system is designed to simulate expert navigation behavior, enabling the neural network to learn and mimic human-like movement and obstacle avoidance. To build the training dataset, the robot is manually controlled in the Gazebo environment for path planning and obstacle avoidance actions. During this process, visual images of the robot are collected as shown in Fig. 7, along with the robot's position, orientation, and movement status as shown in Fig. 8.

By monitoring the robot's motion status, the collected data is divided into two parts. The first part, shown in Fig. 8(a), includes the position data, which consists of the robot's coordinates (x, y) in the simulated world and the robot's vertical axis (Vertical) turning angle z. The second part, shown in Fig. 8(b), represents the motion state, which includes the robot's control outputs: linear velocity υ and angular velocity ω as described in Eq. (4). The range of these velocities is determined by the robot's hardware limitations.

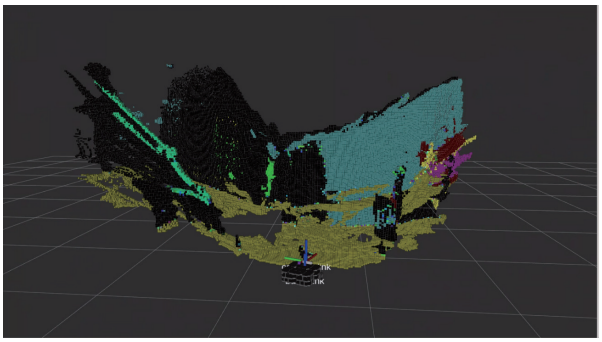


Fig. 5. (Color online) Semantic Octomap.

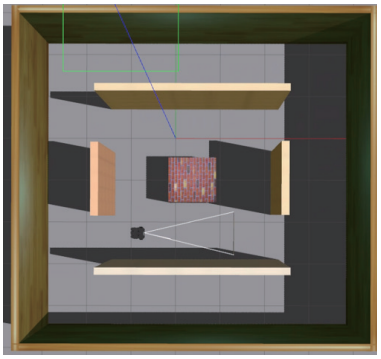
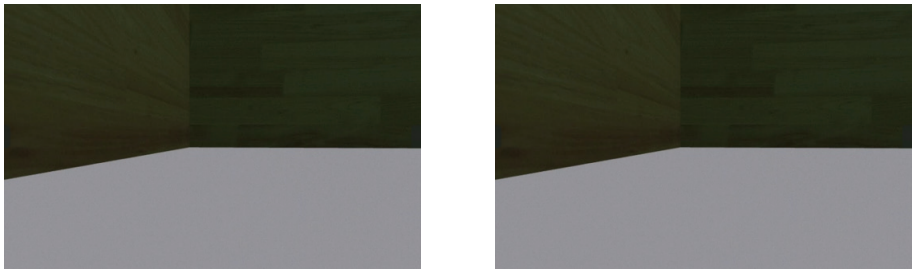
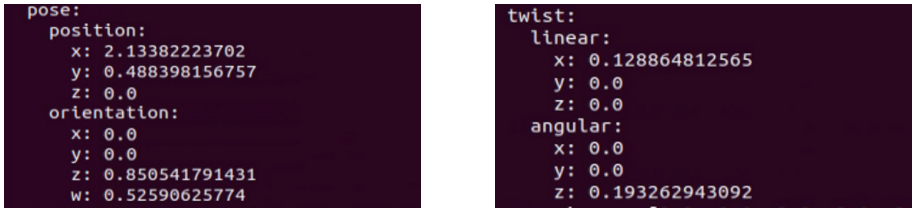


Fig. 6. (Color online) Gazebo environment.



(a) (b)

Fig. 7. (Color online) Image of robot in simulated world. (a) RGB Image. (b) Depth Image.



(a) (b)

Fig. 8. (Color online) Robot position, direction, and motion status. (a) Position and orientation. (b) Linear and angular.

$$v_{el} = \begin{Bmatrix} \upsilon = [v_{min} \sim v_{max}] \ m/s \\ \omega = [\omega_{min} \sim \omega_{max}] \ rad/s \end{Bmatrix} \tag{4}$$

3.3.2 Autonomous navigation neural network

The goal of this paper is to develop a neural network model that can directly predict control commands from color and depth images, achieving an end-to-end autonomous navigation system. This experiment uses EfficientNet[25] as the backbone network to extract meaningful features from composite inputs. The network architecture is shown in Fig. 9.

To train the neural network using visual data and the robot's position and orientation, the network architecture is primarily divided into two parts. The first part uses EfficientNet-B2 as the backbone for feature extraction from images. The second part uses residual modules to extract the robot's coordinates and orientation on the map. The outputs from these components are then combined through fully connected layers to predict the linear velocity υ and the angular velocity ω, achieving end-to-end autonomous navigation. The detailed structure is shown in Table 1.

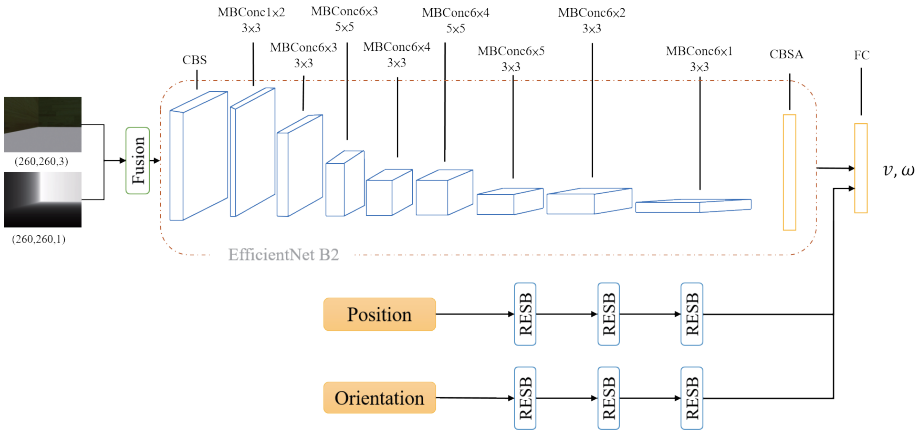


Fig. 9. (Color online) Based on EfficientNet-B2 autonomous navigation neural network.

Table 1
EfficientNet-B2 architecture.

Stage		K	C	R	S	Size
Input						260 × 260 × 3
Layer0	CBS	3	32	1	2	130 × 130 × 32
Layer1	MBConv1	3	16	2	1	130 × 130 × 16
Layer2	MBConv6	3	24	3	2	65 × 65 × 24
Layer3	MBConv6	5	48	3	2	33 × 33 × 48
Layer4	MBConv6	3	88	4	2	17 × 17 × 88
Layer5	MBConv6	5	120	4	1	17 × 17 × 120
Layer6	MBConv6	3	208	5	2	9 × 9 × 208
Layer7	MBConv6	3	352	2	1	9 × 9 × 352
Layer8	CBSA	1	1408	1	1	1 × 1 × 1408
Output	FC		2			υ, ω

The MBConv module, an inverted residual convolutional module, is a key architecture component in the EfficientNet series, as shown in Fig. 10. This module uses depthwise separable convolutions to extract features and integrates inverted residuals. In common residual module practices, images are first downsampled for feature extraction and then upsampled back to the original input dimensions. In contrast, inverted residuals start by upsampling the image to increase the number of channels and expand the feature space, followed by feature extraction and then downsampling to reduce the feature dimensions.

The CBSA structure consists of convolution, batch normalization, SiLU, and global average pooling, as shown in Fig. 11. To train the neural network for robot position and orientation, the RESB residual module is used. The network trains the robot to recognize its position and movement trajectory on the map through coordinates (x, y), which is subsequently used for path planning and tracking. For the direction angle, the network trains the robot to determine the turning angle when encountering obstacles. In this paper, after training with the residual

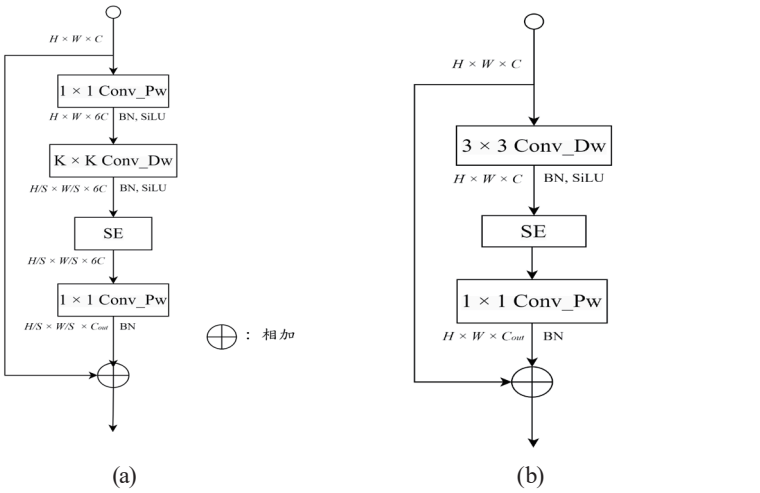


Fig. 10. MBConv module. (a) MBConv6. (b) MBConv1.

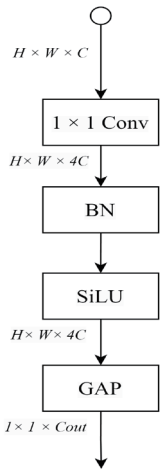


Fig. 11. CBSA module.

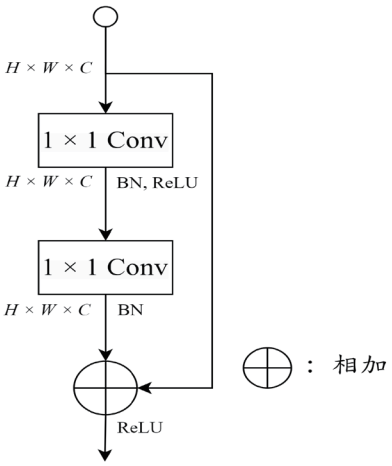


Fig. 12. RESB module.

module, the output is combined with the image neural network's output through fully connected layers to control the robot's speed. The architecture of the residual module is shown in Fig. 12.

In autonomous navigation tasks, since the robot's output linear and angular velocities are continuous and linear, mean squared error (*MSE*) is used as the loss function to quantify the difference between the predicted values and the true values during the training process. *MSE* is commonly used for continuous variable predictions. It is the average of the squared differences between predicted values and true values. Its mathematical formula is given by

$$MSE_{loss} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \widehat{y_i}\right)^2, \tag{5}$$

where n represents the number of samples, y_i is the true value for the i-th sample, and $\widehat{y_i}$ is the model's predicted value for the i-th sample. The error between the true values and the model predictions is squared, summed, and averaged to emphasize the error and improve the model's performance. Adaptive moment estimation (Adam)[26] is used to optimize the network weights, updating the weights based on the gradients of the loss function during backpropagation to minimize the model's loss.

4. Experiment

This paper is divided into two main parts: the first part involves integrating a semantic segmentation network with visual SLAM to create a 3D semantic map, whereas the second part combines deep learning with traditional path planning to achieve an end-to-end autonomous navigation system for robots. The overall architecture is based on the ROS system and utilizes a purely visual approach to implement the functions described in the paper. First, the configuration of the Realsense D435i depth camera is discussed. In the experiments, the resolution of the color and depth images is set to 640 × 480. The data is published in ROS format, including the robot's odometry and the camera's intrinsic parameters. The intrinsic parameters K used in the experiment are shown in Eq. (6).

$$K = \begin{bmatrix} 323.2218 & 0.0 & 605.8509 \\ 0.0 & 242.2031 & 605.4664 \\ 0.0 & 0.0 & 1.0 \end{bmatrix} \tag{6}$$

4.1 Visual semantic segmentation SLAM experimental results

In this section, we will describe the training results of the semantic segmentation neural network based on the edge computing platform, as well as the effectiveness of integrating visual SLAM to generate a 3D semantic Octomap.

4.1.1 Semantic segmentation training results

In this study, considering the robot's movement in complex indoor environments, the neural network was trained using the SUNRGB-D dataset, which includes 5,285 training samples, 5,050 validation samples, and 37 class labels. Figure 13 shows the color correspondence table for all the labels in the SUNRGB-D dataset used in this study.

The neural network architecture proposed in this study was trained on the SUNRGB-D dataset for 200 epochs. The training results are shown in Figs 14–16. Figure 14 presents the accuracy, with the best result being 81.4355%. Figure 15 shows the Mean Intersection over Union (MioU), with the best result being 0.479. Figure 16 shows the loss function, with the best result being 0.2217. Additionally, the network began to converge after approximately 150 epochs of training.

Figure 17 shows the segmentation results of the trained neural network on the SUNRGB-D dataset. (a) and (b) are the input color and depth images, respectively, and (c) shows the model's actual prediction results.

Fig. 13. (Color online) SUBRGB-D dataset semantic tags.

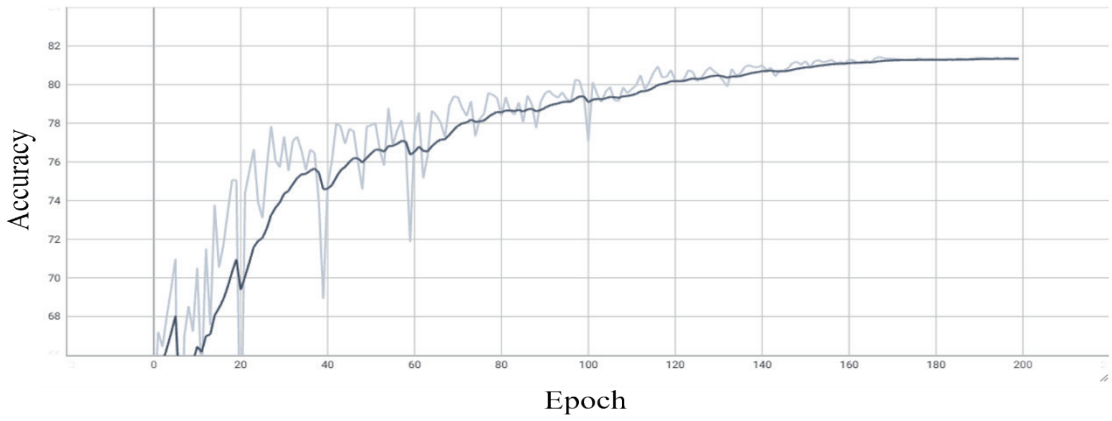


Fig. 14. (Color online) Accuracy.

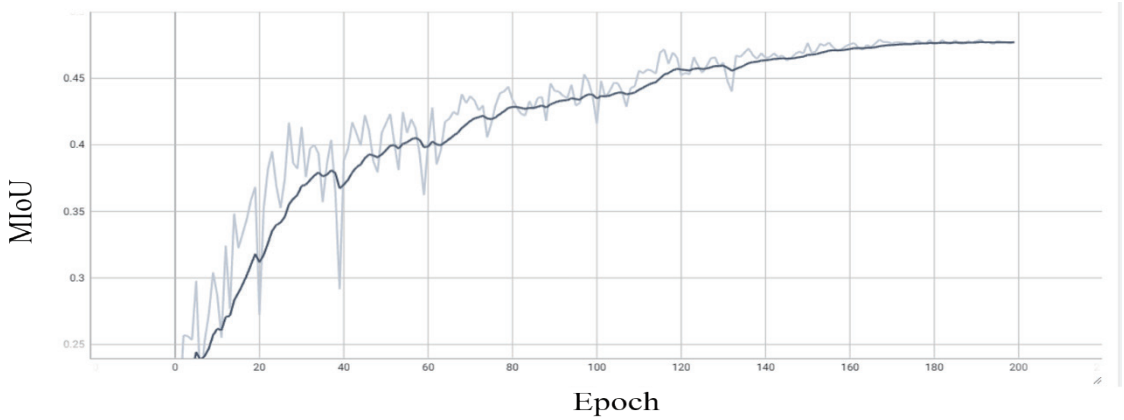


Fig. 15. (Color online) MIoU.

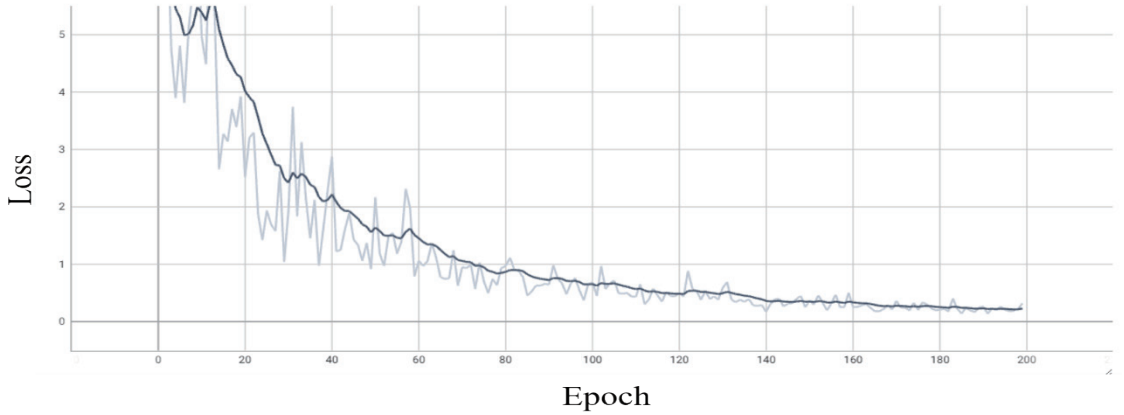
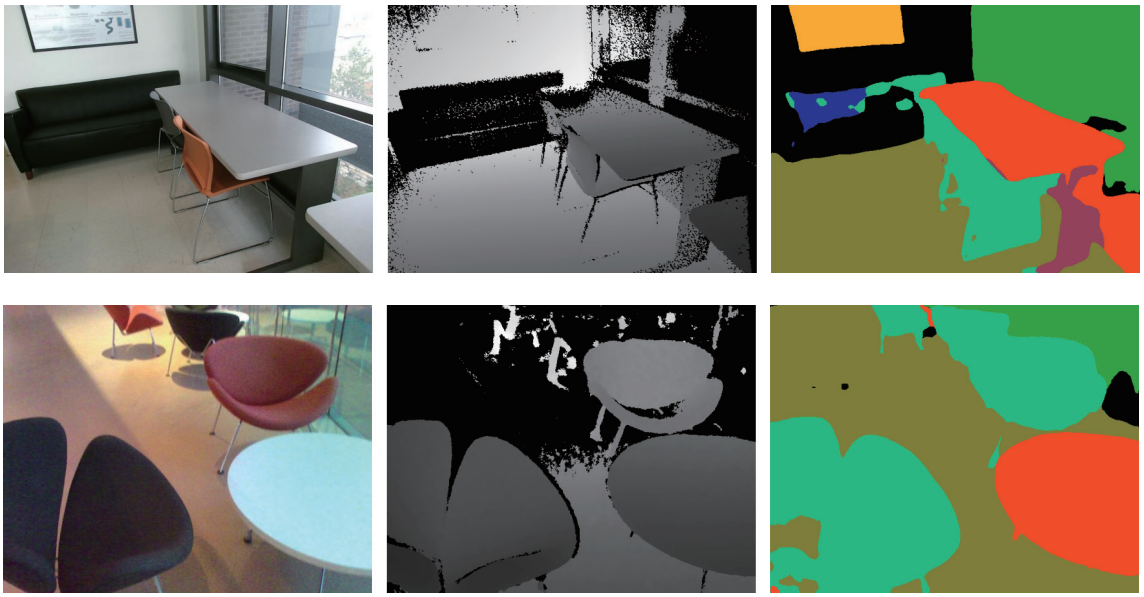


Fig. 16. (Color online) Loss function.



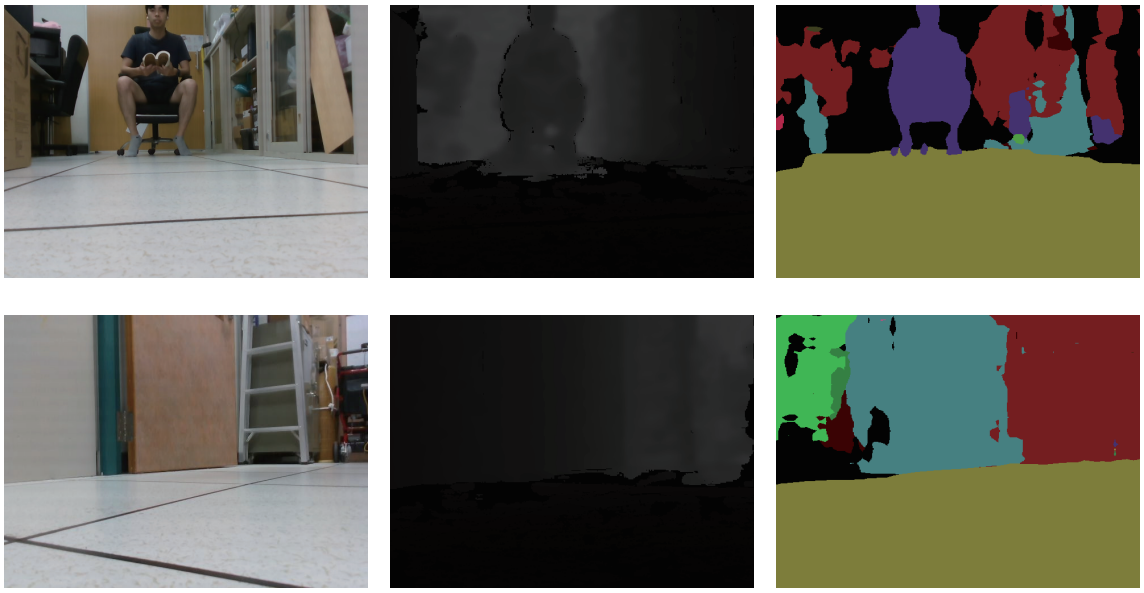
(a) (b) (c)

Fig. 17. (Color online) SUBRGB-D data set segmentation effect. (a) RGB Image. (b) Depth Image. (c) Prediction.

Figure 18 shows the actual segmentation results of the neural network when deployed on a robot. With the AGX and Realsense D435i depth camera, the system performs real-time control output based on each frame of detected images during the robot's dynamic movement, ensuring accuracy and stability in navigation and operation.

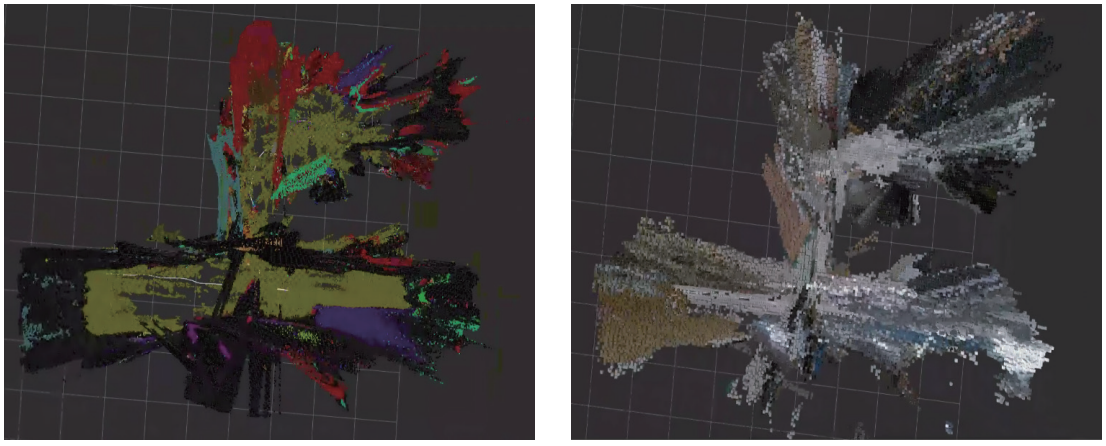
4.1.2 Semantic SLAM experimental results

In this section, we present the results of the semantic SLAM experiments conducted on the Turtlebot3 robot. The semantic segmentation neural network was integrated with visual SLAM to create the Octomap that represents the 3D semantic map, as shown in Fig. 19.



(a) (b) (c)

Fig. 18. (Color online) Visualization of actual segmentation by robots. (a) RGB image. (b) Depth image. (c) Prediction.



(a) (b)

Fig. 19. (Color online) Real world 3D map. (a) Semantic SLAM map. (b) Visual SLAM map.

As shown in Fig 19(a), the colored regions in the image represent the results of the semantic segmentation performed by the robot, which classifies and segments objects, allowing the robot to recognize the surrounding objects. In contrast, Fig 19(b) shows the 3D map generated by visual SLAM. While visual SLAM focuses on reconstructing the geometric structure of the environment and can provide physical structural information, it lacks the capability to identify object types.

4.2 Autonomous navigation neural network experimental results

In Sect. 4.1, we established a static obstacle environment in the Gazebo simulation world and trained the autonomous navigation neural network. The robot, during its movement, needed to avoid obstacles, and the neural network was trained to learn human-like obstacle avoidance behavior. This experiment collected 26062 data points for training and 4000 data points for validation. The training results are shown in Figs. 20–22. The MAE was used as the evaluation metric, demonstrating improvements in the network's prediction accuracy during the learning process. The MAE decreased significantly and stabilized, indicating that the model effectively extracted navigation commands and exhibited good obstacle avoidance capability.

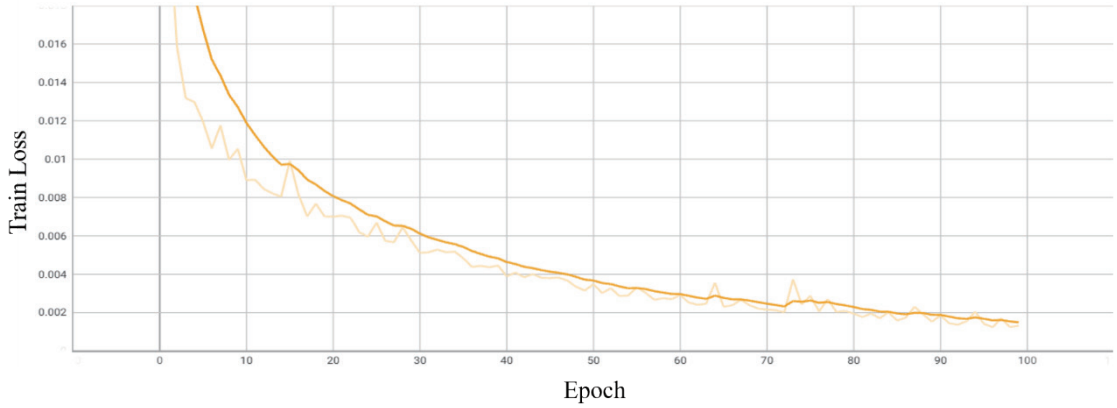


Fig. 20. (Color online) Train Loss Function.

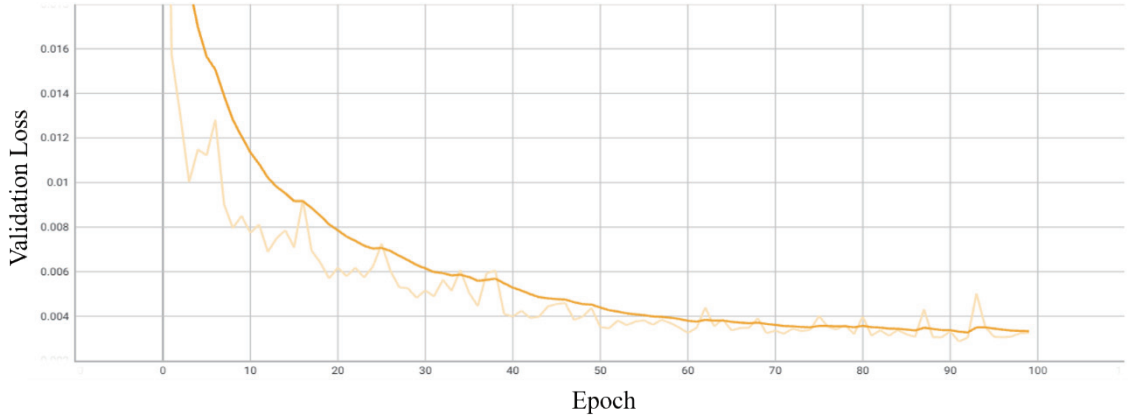


Fig. 21. (Color online) Validation Loss Function.

The experimental results of the autonomous navigation system in real-world navigation are divided into static and dynamic obstacles, presented from a third-person perspective. Figure 23 shows the map created in the real world, with path planning performed using the A-Star algorithm in the Rviz visualization interface. In the figure, the robot model represents the starting point, the red arrow indicates the endpoint, the black lines show the path planned by the A-Star algorithm, the pink-white areas represent the navigable regions, and the black areas denote obstacles.

Figure 24 shows the navigation process with static obstacles from a third-person perspective. The starting point is depicted in the top-left corner of the first image, and the endpoint is shown in the bottom-right corner of the third image. The robot's movement is captured by the camera, demonstrating the robot's navigation trajectory in the real world.

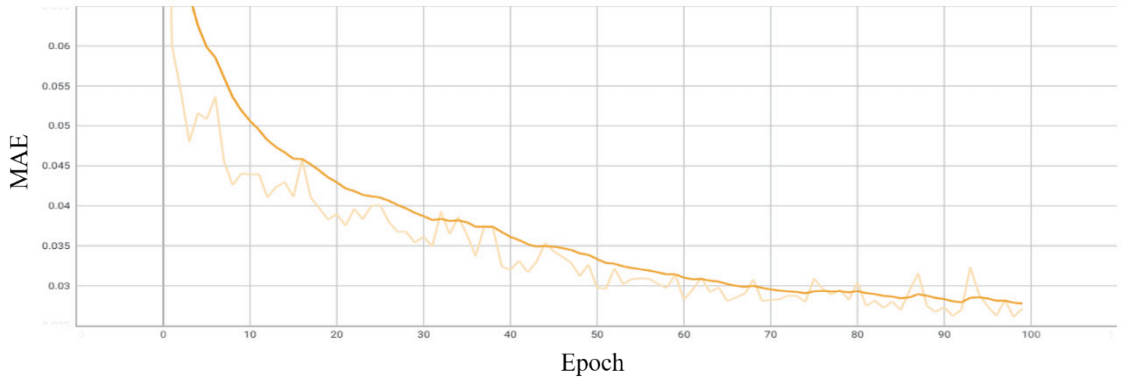


Fig. 22. (Color online) MAE.

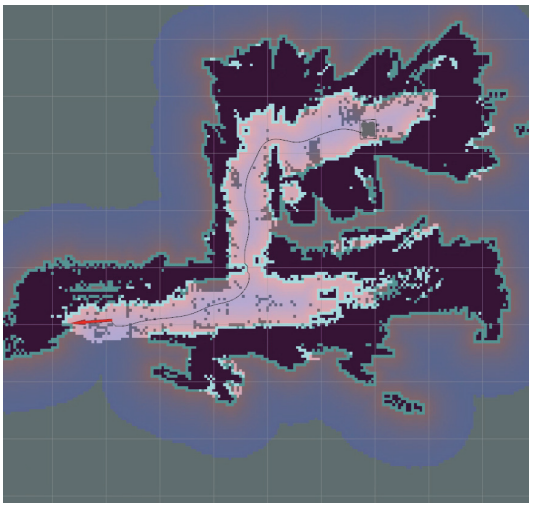


Fig. 23. (Color online) Real world 2D map path planning.

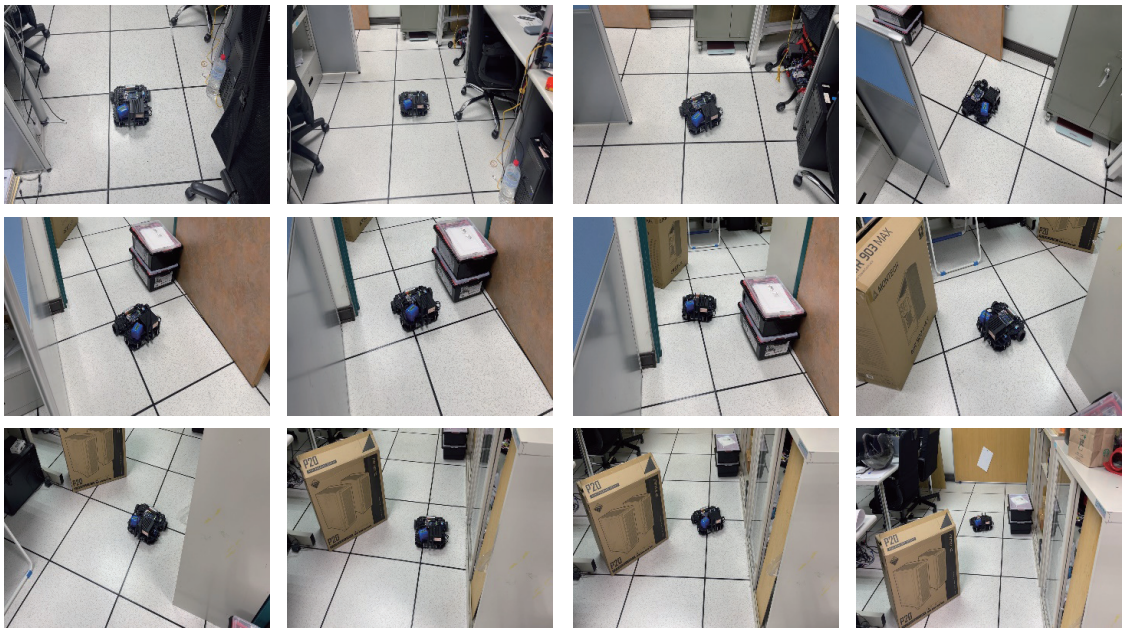


Fig. 24. (Color online) Autonomous navigation neural network static navigation diagram (third person perspective).

5. Conclusion

In this paper, we proposed a purely vision-based robot autonomous navigation system, utilizing deep learning technology to achieve end-to-end navigation. First, semantic segmentation and visual SLAM were combined in the system to construct a three-dimensional environmental map with semantic information, enhancing the robot's understanding and perception of its surroundings. Next, by training a neural network, the system outputted the robot's linear and angular velocities referring to color and depth images, enabling precise autonomous navigation. Finally, the A-Star algorithm was used for path planning, allowing the robot to navigate complex environments and complete indoor autonomous navigation tasks.

Regarding autonomous navigation, the proposed system is currently equipped with a single camera, which restricts perception to the forward-facing field of view. While the robot predominantly moves forward, the lack of lateral perception creates blind spots. Consequently, unexpected events arising outside the camera's coverage may result in irreversible behaviors. To overcome this limitation, in future work, we will consider integrating 4–6 cameras to achieve omnidirectional perception, thereby enhancing robustness and ensuring safer navigation.

References

1 M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba: IEEE Trans. Rob. **17** (2001) 229. https://doi.org/10.1109/70.938381
2 Y. LeCun, Y. Bengio, and G. Hinton: Nature **521** (2015) 436. https://doi.org/10.1038/nature14539
3 Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel: Adv. Neural Inf. Process. Syst. **2** (1989) 396. http://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf

4 Teslas: (2023). https://www.tesla.com/support/transitioning-tesla-vision
5 J. Long, E. Shelhamer, and T. Darrell: Proc. 2015 IEEE Computer Vision and Pattern Recognition Conf. (IEEE, 2015) 3431–3440. https://doi.org/10.1109/CVPR.2015.7298965
6 O. Ronneberger, P. Fischer, and T. Brox: Medical Image Computing and Computer-assisted Intervention–MICCAI 2015: 18th Int. Conf., Munich, Germany, October 5–9, 2015, Proc. Part III (2015) 234. https://doi.org/10.1007/978-3-319-24574-4_28
7 H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia: Proc. 2017 IEEE Computer Vision and Pattern Recognition Conf. (IEEE, 2017) 2881. https://doi.org/10.1109/CVPR.2017.386
8 J. Hu, L. Shen, and G. Sun: Proc. 2018 IEEE Computer Vision and Pattern Recognition Conf. (IEEE, 2018) 7132. https://doi.org/10.1109/CVPR.2018.00745
9 E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo: IEEE Trans. Int. **19** (2017) 263. https://ieeexplore.ieee.org/document/8063438
10 D. Seichter, M. Köhler, B. Lewandowski, T. Wengefeld, and H.-M. Gross: Proc. 2021 IEEE Robotics and Automation Conf. (IEEE, 2021) 13525. https://doi.org/10.1109/ICRA48506.2021.9561782
11 L. C. Chen, G. Papandreou, S. Member,I. Kokkinos, K. Murphy, A. L. Yuille: Trans. Pattern Anal. Mach. Intell. **40** (2017) 834. https://doi.org/10.1109/TPAMI.2017.2699184
12 L. C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam: Proc. 2018 European Conf. Computer Vision Conf. (ECCV, 2018) 801. https://doi.org/10.1007/978-3-030-01261-8_47
13 G. Grisetti, C. Stachniss, and W. Burgard: Trans. Robotics **23** (2007) 34. https://ieeexplore.ieee.org/document/4084563
14 S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf: Proc. 2011 IEEE Symp. Safety, Security, and Rescue Robotics Conf. (IEEE, 2011) 155. https://doi.org/10.1109/SSRR.2011.6106777
15 R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos: Trans. Robotics **31** (2015) 1147. https://doi.org/10.1109/TRO.2015.2463671
16 M. Labbé and F. Michaud: J. Field Robotics **36** (2019) 416. https://doi.org/10.1002/rob.21831
17 Z. Xuan, Filliat: GitHub repository (2018). https://github.com/floatlazer/semantic_slam
18 S. Cheng, C. Sun, S. Zhang, and D. Zhang: Measurement **72** (2022) 1. https://ieeexplore.ieee.org/document/9978699
19 C. Yu, Z. Liu, X. J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei: Proc.2018 IEEE Intelligent Robots and Systems Conf. (IEEE, 2018) 1168. https://arxiv.org/pdf/1809.08379
20 D. A. Pomerleau: Adv. Neural Inf. Process. Syst. **1** (1988) 305. https://proceedings.neurips.cc/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf
21 M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. Jackel, and U. Muller: arXiv:1704.07911 (2017). https://arxiv.org/abs/1704.07911
22 Teslas: (2022). https://www.thinkautonomous.ai/blog/tesla-end-to-end-deep-learning/.
23 S. A. Chen: The Combination of SLAM and Path Planning Technology in the Development of a Mobility Assistive System for Limb Disabled People, Master's Thesis (2020). https://hdl.handle.net/11296/khc85y
24 A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard: Autonomous Robots **34** (2013) 189. https://doi.org/10.1007/s10514-012-9321-0
25 M. Tan and Q. V. Le: arXiv:1905.11946 (2019). https://arxiv.org/abs/1905.11946
26 D. P. Kingma, and J. Ba: arXiv:1412.6980 (2014). https://doi.org/10.48550/arXiv.1412.6980