Sensors and Materials, Vol. 38, No. 5 (2026) 2661–2680
MYU Tokyo

S & M 4461

A Multimodule Navigation Platform for Smart Wheelchairs Integrating Relation Transformer and Semantic Mapping

Neng-Sheng Pai, Chi-Jui Wu, Pi-Yun Chen,* and Xiang-Yan Tsai

Department of Electrical Engineering, National Chin-Yi University of Technology,
Taichung City 41170, Taiwan (ROC)

(Received December 19, 2025; accepted April 6, 2026)

Keywords: smart wheelchair, semantic SLAM mapping, visual relationship detection, path planning and navigation, TEB local planner, relation transformer

In this study, we extend the development of the laboratory's smart wheelchair system by proposing a semantically aware solution tailored for elderly individuals with visual impairments and physical disabilities, aiming to alleviate the shortage of caregiving manpower in an aging society. The system integrates three core modules: semantic simultaneous localization and mapping (SLAM), path planning and navigation, and visual relationship detection (VRD). With an RGB-D camera sensor, the system performs oriented fast and rotated brief (ORB)-SLAM for localization and mapping, then applies semantic segmentation to label the point cloud and construct a 3D semantic map. For navigation, the Timed Elastic Band (TEB) algorithm is adopted to dynamically avoid obstacles and generate feasible paths. In terms of scene understanding, the system incorporates the Relation Transformer (RelTR) model to identify semantic relationships between objects and provides real-time feedback to the user through voice prompts, enhancing spatial awareness and decision-making. The complete system was integrated into a physical wheelchair for testing. Experimental results demonstrate its capability in stable mapping, effective obstacle avoidance, and semantic comprehension, contributing to greater mobility for visually impaired individuals while reducing caregiver workload.

1. Introduction

With advances in medical technology, Taiwan has entered an aging society. According to the World Health Organization, when the proportion of the population aged 65 and over reaches 7, 14, and 20%, the society is classified as aging, aged, and super-aged, respectively.(1) As of 2024, the proportion of elderly people aged 65 and over in Taiwan has reached 19.18%, indicating that the country is on the verge of becoming a super-aged society.(2)

As a result, the demand for long-term care in Taiwan has increased accordingly. As of May 2023, approximately 842000 elderly people in Taiwan required long-term care, while only around 95000 caregivers were officially registered, indicating an imbalance between care demand and workforce supply.(3)

*Corresponding author: e-mail: chenby@ncut.edu.tw
https://doi.org/10.18494/SAM6130

ISSN 0914-4935
https://myukk.org/

Among them, elderly individuals with visual impairment face greater difficulties in mobility. Studies have shown that common causes include cataract, age-related macular degeneration (ARMD), diabetic retinopathy, glaucoma, and other retinal and corneal diseases.(4)

To improve mobility challenges faced by this population, we propose in this paper an intelligent mobility assistive system that integrates semantic simultaneous localization and mapping (SLAM), path planning, and visual relationship detection (VRD). The system combines semantic segmentation-based mapping, dynamic navigation, and relational reasoning capabilities to provide vision-guided environmental awareness and safe mobility support, aiming for application in smart elderly care. In this paper, we will next conduct a literature review on three key components: SLAM, path planning and navigation, and VRD.

1.1 SLAM

Since the 1980s, SLAM technology has continued to evolve to meet the challenges of complex environments. Early approaches such as Extended Kalman Filter SLAM (EKF-SLAM) achieved SLAM for the first time by linearizing motion and observation models. However, owing to high computational costs, it was only suitable for small-scale environments.(5) In 2000, Montemerlo *et al.* proposed Fast SLAM, which decouples pose estimation and map construction by integrating particle filtering with Kalman filters, considerably enhancing system scalability.(6) In 2015, Mur-Artal *et al.* introduced oriented fast and rotated brief (ORB)-SLAM, which utilizes ORB features, loop closure, and global optimization to enhance stability and accuracy, becoming a representative work in modern SLAM systems.(7)

Further development of semantic SLAM builds upon ORB-SLAM by incorporating semantic segmentation models, enabling the map to include object category information and enhancing the robot's semantic perception and scene understanding capabilities.(8)

1.2 Path planning and navigation

Path planning is a core technology that enables a robot to safely navigate from a starting point to a target location. The A* (A-star) algorithm, proposed by Hart *et al.* in 1968, is capable of efficiently finding the shortest path in static environments.(9) To handle dynamic environments, Stentz proposed the D* (Dynamic A-star) algorithm, which reduces computational load by performing localized updates.(10) Subsequently, Koenig and Likhachev developed D* Lite, which further simplified the algorithm's structure and improved its computational efficiency.(11)

To address dynamic obstacle environments, Fox *et al.* proposed the Dynamic Window Approach (DWA), which generates control commands in real time on the basis of the robot's dynamic constraints, enabling flexible obstacle avoidance.(12) However, DWA is prone to getting stuck in local minima or exhibiting oscillatory behavior. To address this issue, Rösmann *et al.* proposed the Timed Elastic Band (TEB) in 2015, which integrates nonlinear optimization with dynamic constraints to generate smooth and safe trajectories through joint temporal and spatial optimization.(13) TEB is particularly well suited for high-obstacle-density and dynamic environments, making it a widely adopted advanced local planner in modern smart wheelchairs and autonomous driving systems.

1.3 VRD

VRD aims to recognize objects in an image and identify their semantic relationships, typically represented in the form of subject–predicate–object, such as "person–rides–horse". This technique enhances the semantic understanding of images and is widely applied in image retrieval, robotic vision, and visual question answering (VQA).

The novel Relation Transformer (RelTR)[14] was designed on the basis of the Detection Transformer (DETR) architecture.[15] It bypasses the conventional two-stage process of "detecting objects first, then predicting relationships", and instead directly performs the end-to-end prediction of a fixed number of semantic triplets. RelTR introduces three key modules: Coupled Self-Attention (CSA), Decoupled Visual Attention (DVA), and Decoupled Entity Attention (DEA), which together enhance prediction efficiency and accuracy. The model has demonstrated outstanding performance on datasets such as Visual Genome, VRD, and OpenImages-V6.

VRD techniques have evolved from traditional rule-based methods to unified reasoning frameworks driven by deep learning and Transformer architectures, demonstrating powerful semantic understanding and broad application potential.

2. Methodology

The proposed system is built upon the Merits P113 YOYO foldable electric wheelchair as the base platform,[16] with further optimization and integration applied. The overall system is divided into three main modules: Semantic SLAM, path planning and navigation, and VRD. The system architecture is illustrated in Fig. 1.

2.1 Semantic SLAM

The Semantic SLAM system consists of four key modules: semantic segmentation, SLAM, semantic point cloud generation, and octree structure. First, the semantic segmentation module

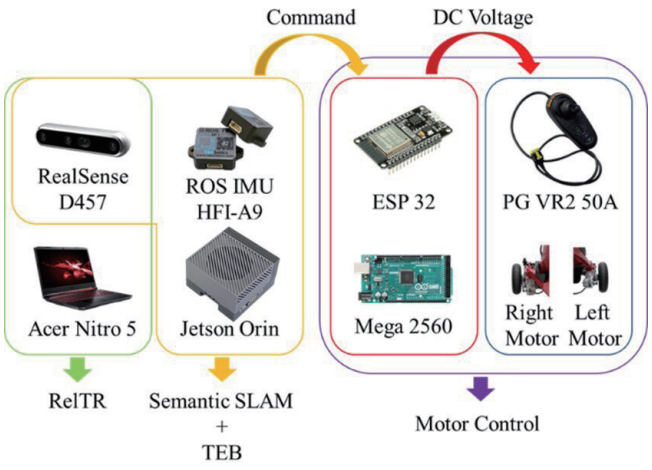


Fig. 1. (Color online) Smart wheelchair system architecture diagram.

uses deep learning models to perform pixel-wise classification on images, enhancing object-level understanding. Then, the SLAM system, based on ORB-SLAM, integrates semantic and geometric information to achieve real-time localization and mapping. The semantic point cloud module combines semantic labels with 3D spatial positions to construct a semantic point cloud. Finally, the octree module compresses the point cloud data while preserving semantic information, facilitating efficient map storage and retrieval. The overall architecture is shown in Fig. 2.

In this study, we adopted the Pyramid Scene Parsing Network (PSPNet)[17] as the core enhancement module for the semantic segmentation neural network to improve scene understanding and the accuracy of semantic labeling. The key component of PSPNet is the Pyramid Pooling Module (PPM), which performs feature pooling at multiple scales to capture both global and local semantic information from the image. This enables the model to effectively handle large scenes and complex backgrounds.

PSPNet offers two major advantages: First, its PPM captures image features at multiple scales simultaneously, enhancing labeling accuracy for objects of various sizes and distances. Second, by fusing features across multiple layers, PSPNet improves the recognition of object boundaries and fine details, making it especially suitable for scenes with diverse objects and complex edges. As shown in Fig. 3, the overall architecture takes an input image and extracts features via a convolutional neural network. These features undergo multiscale pooling and upsampling, then are concatenated along the channel dimension to form a composite feature map. This produces semantic representations with both global context and detailed perception. Owing to its high efficiency and accuracy, PSPNet is an ideal choice for the semantic segmentation task in semantic SLAM, providing a stable and precise semantic foundation for subsequent 3D mapping and robotic decision-making.

ORB-SLAM serves as the core geometric mapping module. It performs real-time localization and mapping based on keypoint features, offering high accuracy and robust loop closure detection. The framework is illustrated in Fig. 4.

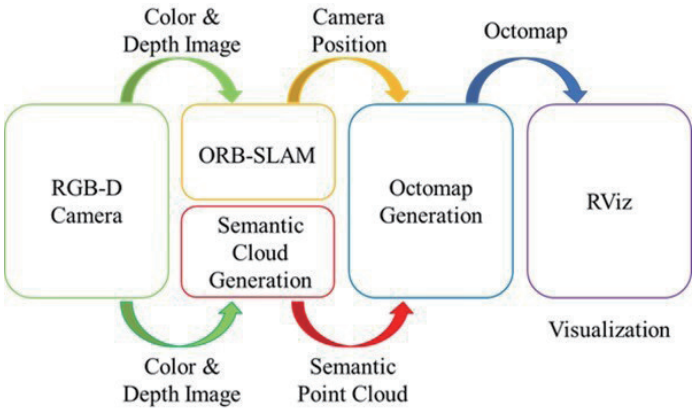


Fig. 2. (Color online) Semantic SLAM architecture diagram.

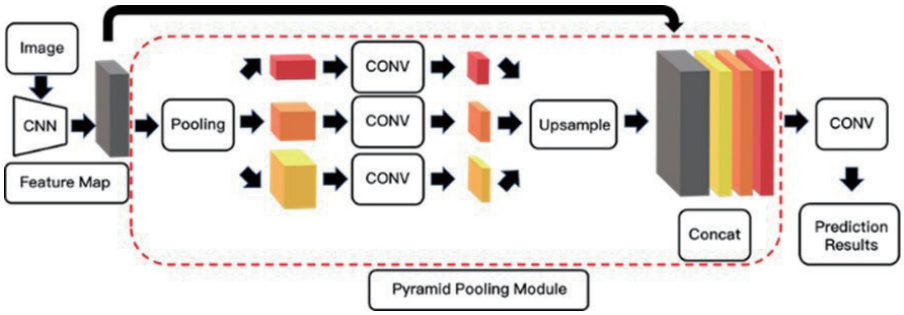


Fig. 3. (Color online) PSPNet architecture diagram.

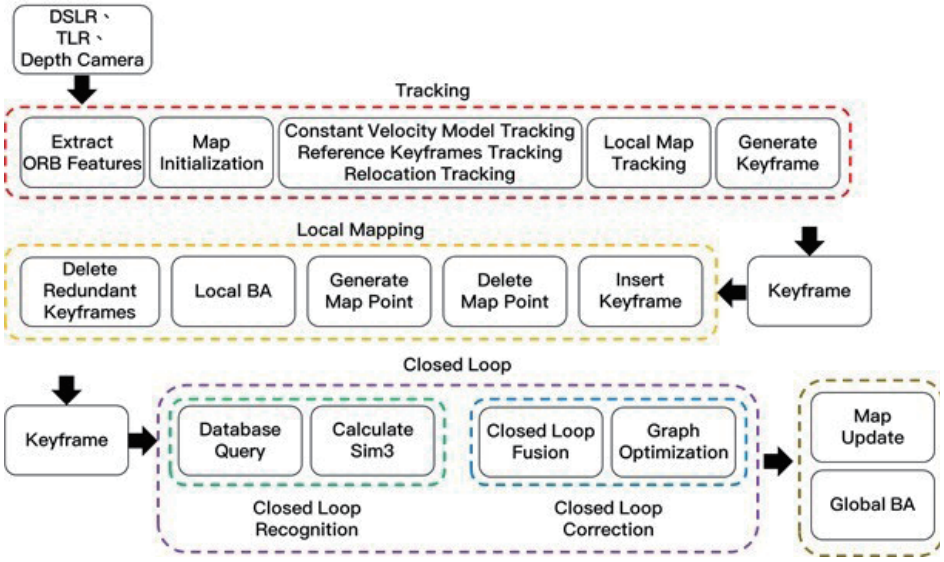


Fig. 4. (Color online) ORB-SLAM frame diagram.

This system utilizes the RealSense D457 depth camera to capture RGB-D images, from which ORB features are extracted for each frame to perform keyframe selection and pose estimation. ORB features offer advantages such as fast computation and invariance to scale and rotation, serving as the foundation for subsequent tracking and matching. ORB-SLAM employs three tracking modes: motion-only tracking between consecutive frames, local map tracking, and relocalization. Camera pose is computed by minimizing the reprojection error, as shown in Eq. (1).[7]

$$e_i = x_i - \pi\left(TX_i\right) \tag{1}$$

Here, e_i represents the observation error, x_i is the image coordinate of the feature point, $\pi(\cdot)$ denotes the projection function, T is the camera pose, and X_i is the 3D coordinate of the feature point. When a new keyframe is detected, the system performs bundle adjustment (BA) to optimize all feature points and camera poses by minimizing the sum of squared reprojection errors, as shown in Eq. (2).[7]

$$\min \sum_{i,j} \left\| X_i^j - \pi(T_j X_i) \right\|^2 \tag{2}$$

In the semantic mapping process, the geometric map generated by ORB-SLAM is augmented with semantic annotations. The semantic segmentation network classifies each pixel in the RGB image and, combined with the corresponding depth image, projects the semantic labels into 3D space to form a semantic point cloud. The 3D coordinates are obtained using Eq. (3).[8]

$$z = D(u,v),\ x = \frac{(u - C_x) \cdot z}{f_x},\ y = \frac{(v - C_y) \cdot z}{f_y} \tag{3}$$

Here, $D(u, v)$ represents the depth of the pixel, whereas C_x, C_y and f_x, f_y are the intrinsic parameters of the camera. The resulting output is a 3D point cloud enriched with semantic labels, serving as the foundation for the semantic SLAM map and supporting subsequent robot decision-making and navigation.

To efficiently represent the semantic point cloud, we adopted Octomap[18] as the spatial mapping framework. Octomap is based on a recursive octree structure that hierarchically divides the 3D space into 1/8-sized volumes. Each voxel is labeled as "occupied", "free", or "unknown", and the framework supports multiresolution representations. This sparse mapping structure is well suited for large-scale environments, offering excellent query efficiency and scalability.

After generating the semantic point cloud, the system projects the points into the corresponding voxels in the Octomap based on the camera pose information and updates their semantic attributes. To integrate multiple observations, the fusion of semantic labels adopts either a Max Fusion or Bayesian Fusion strategy. Octomap is originally designed as a Probabilistic Occupancy Grid, and its state update follows Bayesian filtering theory, as expressed in Eq. (4).[19]

$$P(m \mid z_{1:t}) = \frac{P(z_t \mid m) \cdot P(m \mid z_{1:t-1})}{P(z_t \mid z_{1:t-1})} \tag{4}$$

Here, $P(m|z_{1:t})$ denotes the posterior probability that a voxel m is occupied after the tth observation, $P(z_t|m)$ is the observation model, $P(m|z_{1:t-1})$ represents the prior probability, and $P(z_t|z_{1:t-1})$ is the normalization term.

In the semantic domain, Bayesian fusion is likewise applicable. When a voxel receives semantic observations Z from multiple frames, the confidence update for its semantic label S follows the formula shown in Eq. (5).[19]

$$p(S \mid Z) = \frac{p(Z \mid S) \cdot p(S)}{p(Z)} \tag{5}$$

Here, $P(S|Z)$ denotes the posterior probability that the voxel belongs to the semantic category S given the observation Z, $p(Z|S)$ represents the likelihood of the observation given the label, $p(S)$ is the prior probability of the label, and $p(Z)$ is the marginal probability used for normalization.

Through the above method, each voxel contains not only geometric occupancy information but also semantic labels with associated confidence levels. This enables the semantic map to exhibit consistency and reasoning capability, allowing it to be further applied to tasks such as navigation and scene understanding.

2.2 Path planning and navigation

The navigation system in this paper integrates global and local path planning to ensure the stable movement of the smart wheelchair in complex environments. For global planning, the Dijkstra algorithm is adopted; although slightly less efficient than A*, it guarantees the shortest path and is suitable for static maps. To handle dynamic obstacles, the system employs TEB as the local planner, offering real-time obstacle avoidance and path optimization. Parameters are tuned to fit the wheelchair's motion characteristics, enabling smooth navigation in narrow or changing environments. Compared with DWA, TEB considers both temporal and spatial factors, delivering more flexible and accurate performance.

TEB is a local path planning method based on graph optimization. It models the robot's motion trajectory as a discrete spatiotemporal path composed of a series of timestamped pose nodes $X_i = (x_i, y_i, \theta_i, t_i)$. These nodes collectively form the full trajectory $X = \{X_1, X_2, ..., X_N\}$. By optimizing both the spatial positions and temporal parameters of these nodes, TEB enables real-time obstacle avoidance and smooth navigation in dynamic environments. The architecture of TEB is illustrated in Fig. 5.

The core process of TEB can be divided into the following steps:

(1) Initialization and Discretization of the Path

The initial path is obtained from the global planner (Dijkstra) and discretized into spatiotemporal nodes to form an initial elastic band.

(2) Hypergraph Construction and Graph Optimization

TEB transforms the entire trajectory into a hypergraph structure, where each node represents a pose-time pair, and the edges encode motion and environmental constraints (such as obstacle avoidance, velocity, and acceleration). Using the g2o graph optimization framework, TEB minimizes the total energy function composed of various cost terms, as shown in Eq. (6).[13]

$$J = \omega_p J_p + \omega_t J_t + \omega_s J_s + \omega_o J_o + \omega_d J_d \tag{6}$$

Here, J_p represents the path length (encouraging shorter paths), J_t enforces temporal smoothness (controlling velocity changes), J_s encourages turning smoothness (reducing curvature fluctuations), J_o is the obstacle cost (exponentially increasing with proximity to obstacles), and J_d represents kinematic constraints (including velocity and acceleration limits). ω_p, ω_t, ω_s, ω_o, and ω_d are the weight coefficients corresponding to each cost term.

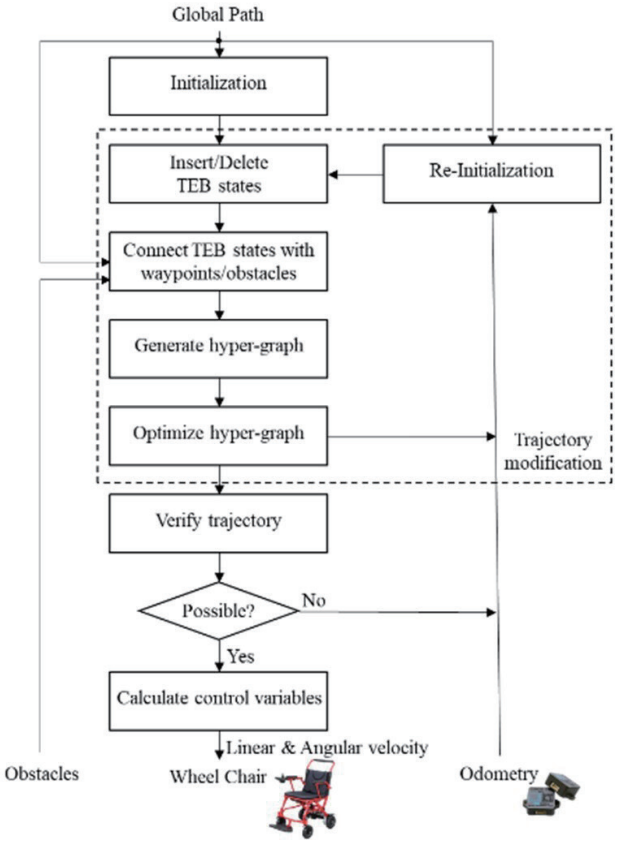


Fig. 5. (Color online) TEB architecture diagram.

They are used to adjust the relative importance of each component in the overall energy function.

(3) Time Modeling and Dynamic Adaptation

TEB adjusts the time interval Δt_i between nodes to control velocity and acceleration, thereby achieving temporal flexibility—particularly suitable for reactive navigation in dynamic environments. To ensure kinematic feasibility, TEB imposes constraints on time, velocity, and acceleration, as shown in Eq. (7).[13]

$$v_{min} \leq \frac{d(X_i, X_{i+1})}{t_{i+1} - t_i} \leq v_{max} \tag{7}$$

Here, $d(X_i, X_{i+1})$ denotes the spatial distance between two consecutive nodes X_i and X_{i+1}, while $t_{i+1} - t_i$ represents the corresponding time interval. This equation enforces that the average velocity between consecutive nodes must remain within an acceptable range, specifically between the minimum velocity v_{min} and the maximum velocity v_{max}.

(4) Obstacle Avoidance and Boundary Control

To ensure safety, TEB imposes a distance constraint between each node and nearby obstacles, as shown in Eq. (8).[13]

$$d(X_i, O_j) \geq d_{safe} \tag{8}$$

Here, $d(X_i, O_j)$ denotes the Euclidean distance between the node X_i and the obstacle O_j, while d_{safe} represents the safety distance threshold, which defines the minimum allowable distance between the smart wheelchair and any obstacle.

(5) Control Output and Trajectory Validation

If the optimized trajectory satisfies all constraints, it is converted into control commands (linear and angular velocities). Otherwise, the trajectory is reinitialized and iteratively optimized until a feasible solution is obtained.

(6) Graph Optimization Objective Function Equation

All costs and constraints are ultimately formulated as a nonlinear least squares problem, which is solved using the g2o optimization framework, as shown in Eq. (9).[13]

$$X^* = \arg\min_X \sum_i e_i(X)^T W_i e_i(X) \tag{9}$$

Here, $e_i(X)$ represents each cost or error function and W_i is the corresponding weight.

2.3 VRD

The RelTR model used in this paper is an end-to-end VRD model built upon DETR, capable of directly predicting ⟨subject–predicate–object⟩ triplets from images. Unlike traditional two-stage approaches that first detect objects and then classify relationships between object pairs, RelTR adopts a query-based design that unifies object recognition and semantic pairing within a single model, thereby simplifying the pipeline and improving inference efficiency. The DETR architecture shown in Fig. 6 encodes image features with positional embeddings and feeds them into a Transformer encoder. Query vectors then interact with these features through the decoder via cross-attention to learn the spatial and semantic information of objects, ultimately predicting object categories and bounding boxes.

The architecture of RelTR, as illustrated in Fig. 7, extends DETR by dividing the query vectors into Subject Queries and Object Queries. These are individually processed through an Entity Decoder to predict the corresponding subjects and objects in the image, outputting their

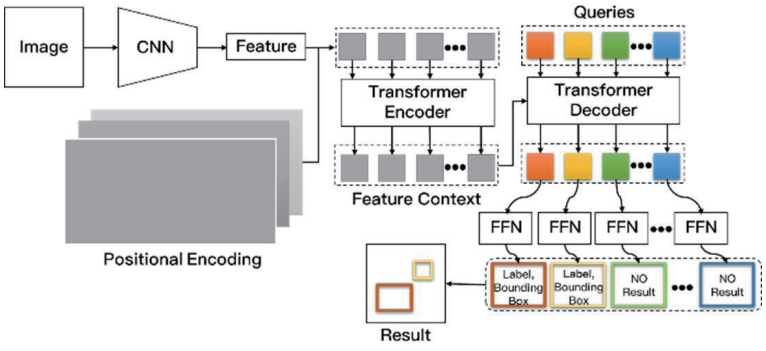


Fig. 6. (Color online) DETR architecture diagram.

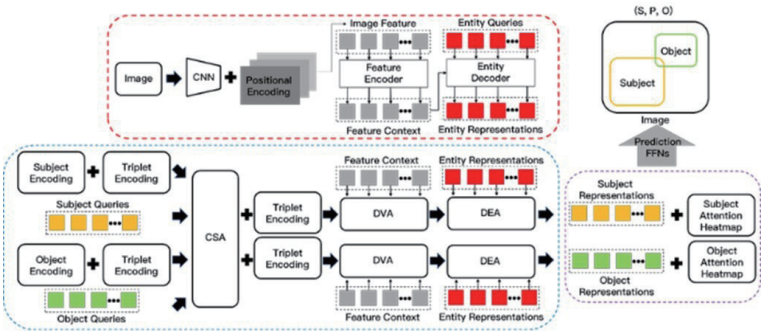


Fig. 7. (Color online) RelTR architecture diagram.

categories and bounding boxes. Each subject-object query pair is then fed into the Triplet Decoder module, where it passes through three attention mechanisms—Cross Structure Attention (CSA), Dynamic Visual Aggregator (DVA), and Dynamic Entity Aggregator (DEA)—to enable interaction and semantic fusion. CSA enhances semantic alignment between subject and object queries, DVA aggregates relevant regional information from the image features, and DEA further integrates visual and query semantics to produce the final relational semantic representation. The predicted triplet is then passed through a feed-forward neural network to output the predicate class, which is matched against ground-truth relationship annotations for training.

To achieve one-to-one prediction matching, RelTR adopts the Hungarian matching algorithm, the same as in DETR, to establish the optimal correspondence between query outputs and ground-truth annotations. The overall loss function is designed to account for three subtasks: subject detection, object detection, and relationship prediction. The total loss can be expressed as Eq. (10).[14]

$$L_{total} = \lambda_s L_{sub} + \lambda_o L_{obj} + \lambda_r L_{rel} \tag{10}$$

Here, L_{sub} and L_{obj} represent the losses for the detection of the subject and object, respectively, whereas L_{rel} denotes the loss for relationship classification. The coefficients λ_s and λ_o are the weights for subject–object detection and relationship prediction losses, respectively, used to balance their contributions to the total loss.

Both the subject and object losses consist of classification and bounding box regression losses, as defined in Eq. (11).[14]

$$L_{sub} = L^s_{cls} + L^s_{box}, L_{obj} = L^o_{cls} + L^o_{box} \tag{11}$$

The classification losses for the subject and object are denoted as L^s_{cls} and L^o_{cls}, whereas the bounding box regression losses for the subject and object are represented as L^s_{box} and L^o_{box}, respectively.

The relationship classification loss predicts the predicate for each subject–object query pair. To address the issue of excessive background samples, a Weighted Cross-Entropy Loss is applied, as defined in Eq. (12).[14]

$$L_{rel} = \frac{1}{|P|} \sum_{(s_i, o_j) \in P} w_{y_{ij}} \cdot CE\left(\hat{y}_{ij}, y_{ij}\right) \tag{12}$$

Here, P denotes the set of all subject–object query pairs, where (s_i, o_j) represents the ith subject and jth object query pair. y_{ij} is the ground truth predicate label and $\hat{y}_{ij}$ is the predicted probability distribution over predicate classes for that pair. $CE(\hat{y}_{ij}, y_{ij})$ denotes the cross-entropy loss and $w_{y_{ij}}$ is a weighting factor for the corresponding class, used to address class imbalance.

Through the above design, RelTR not only features a concise architecture and strong interpretability but also effectively learns structured semantic relationship triplets without the need for additional candidate box generation or postprocessing mechanisms. This results in a highly integrated and efficient inference pipeline, offering a scalable and end-to-end solution for VRD tasks.

3. Experiments

3.1 Semantic SLAM

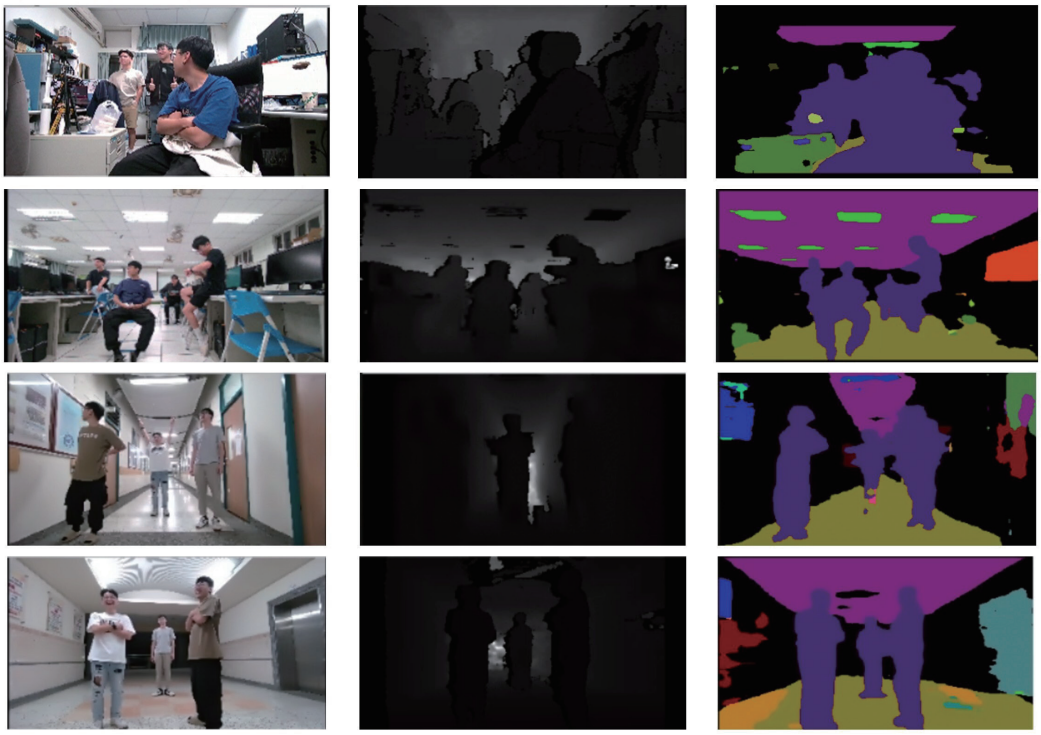
In this study, we adopted PSPNet as the primary architecture for semantic segmentation, integrating a ResNet-50 backbone with a PPM, offering excellent multiscale semantic parsing capability. Table 1 provides a detailed description of the PSPNet layer structure, where feature map dimensions are expressed in the format $H \times W \times C$. The convolutional parameters—kernel size (k), the number of channels (c), stride (s), and dilation rate (r)—along with the number of output classes (C), are clearly listed to illustrate the model architecture.

We utilized the ADE20K public dataset for model training. Figure 8 illustrates the semantic segmentation results of the model in various scenes, including laboratory and campus environments. The visualizations include (a) the original RGB image, (b) the depth map, and (c) the corresponding semantic segmentation map.

To implement semantic SLAM, in this study, we integrated a semantic segmentation model and the Octomap framework based on ORB-SLAM2 to construct a semantic 3D map. The system setup process includes installing Pangolin 0.5 and OpenCV 2.4.3, compiling the ROS

Table 1
Semantic segmentation training network architecture table.

Stage	Structure definition				Output size
	k	c	s	r	
Input		–			(473 × 473 × 3)
Conv1	7	64	2	–	(237 × 237 × 64)
MaxPool	3	–	2	–	(118 × 118 × 64)
Conv2_x	3	256	1	3	(118 × 118 × 256)
Conv3_x	3	512	2	4	(59 × 59 × 512)
Conv4_x	3	1024	2	6	(30 × 30 × 1024)
Conv5_x	3	2048	2	3	(15 × 15 × 2048)
PPM		pool=[1 × 1, 2 × 2, 3 × 3, 6 × 6], Concat			(15 × 15 × 4096)
Bottleneck	1	512	–	–	(15 × 15 × 512)
Classifier	1	$c = C$	–	–	(15 × 15 × C)
Upsample		Scale = 32			(473 × 473 × C)



(a) (b) (c)

Fig. 8. (Color online) Semantic segmentation model results. (a) Original, (b) depth, and (c) semantic images.

version of ORB-SLAM2, installing the RealSense driver, and launching rs_rgbd.launch to acquire RGB-D data. After setup, camera intrinsics and distortion parameters must be configured (as shown in Table 2), along with specifying the semantic model and image topic, and adjusting the Octomap resolution and generation parameters.

The system execution process consists of four steps: launching the camera driver, starting the ORB-SLAM2 node for real-time localization, activating the semantic mapping node to generate semantic point clouds and perform Octomap fusion, and simultaneously using RViz for real-time visualization. Figure 9 illustrates the overall mapping process, showing visual results at the initial (a), intermediate (b), and completed (c) stages, including feature point distribution, 3D map structure, and semantic annotations.

The final semantic mapping results are shown in Fig. 10, which presents the 3D semantic map constructed using the ADE20K model and visualized from multiple perspectives to highlight its semantic structure. Figure 11 shows the 2D map generated by the traditional ORB-SLAM system, which serves as the geometric basis for navigation. The overall results demonstrate that the proposed system can stably perform semantic mapping in indoor environments, effectively integrating semantic information with geometric structures.

3.2 Navigation and obstacle avoidance

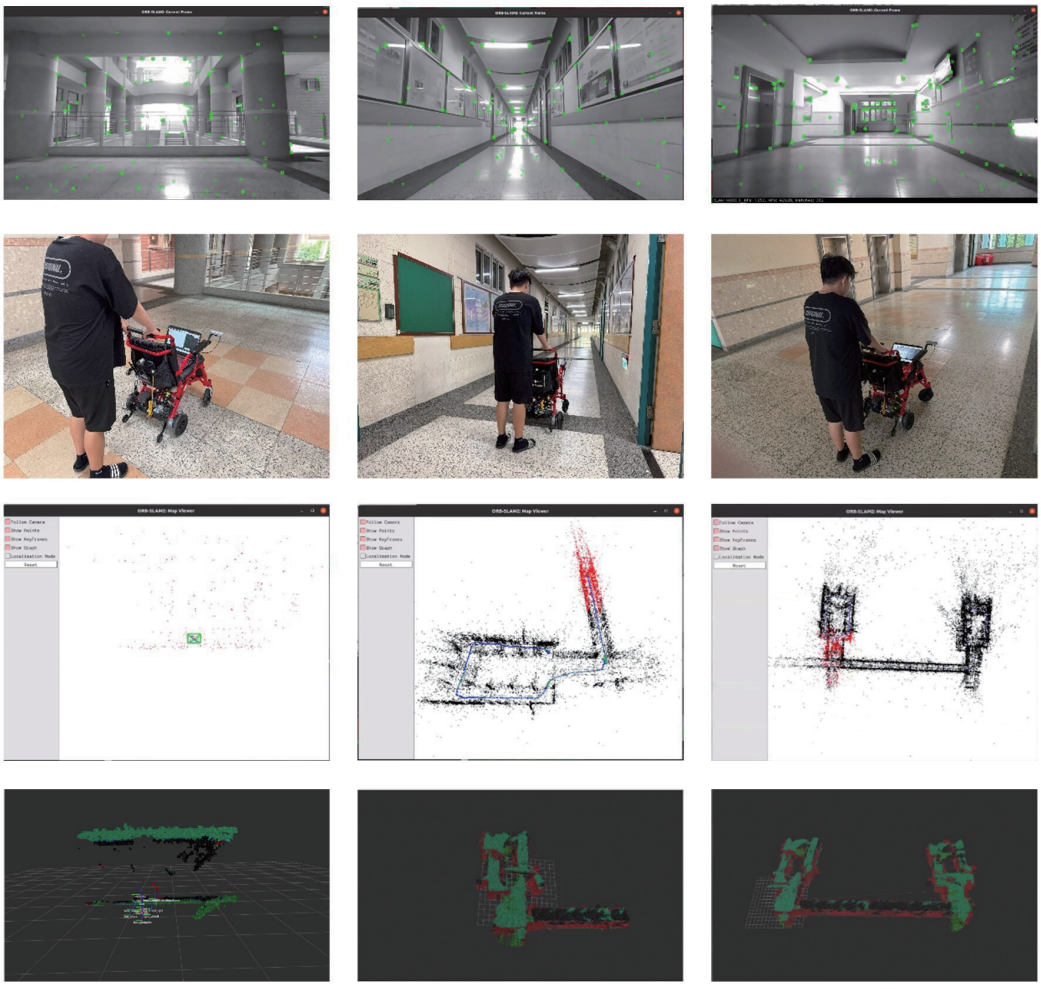
To enable the real-time detection and avoidance of dynamic obstacles, we designed an obstacle estimation pipeline that integrates YOLOv5[20] object detection with depth sensing. The system first detects human targets in the image using YOLOv5 and obtains the bounding box in

Table 2
RealSense D457 camera parameter table.

%YAML: 1.0
Camera.fx: 643.891357421875
Camera.fy: 642.9199829101562
Camera.cx: 650.5823974609375
Camera.cy: 366.5848388671875

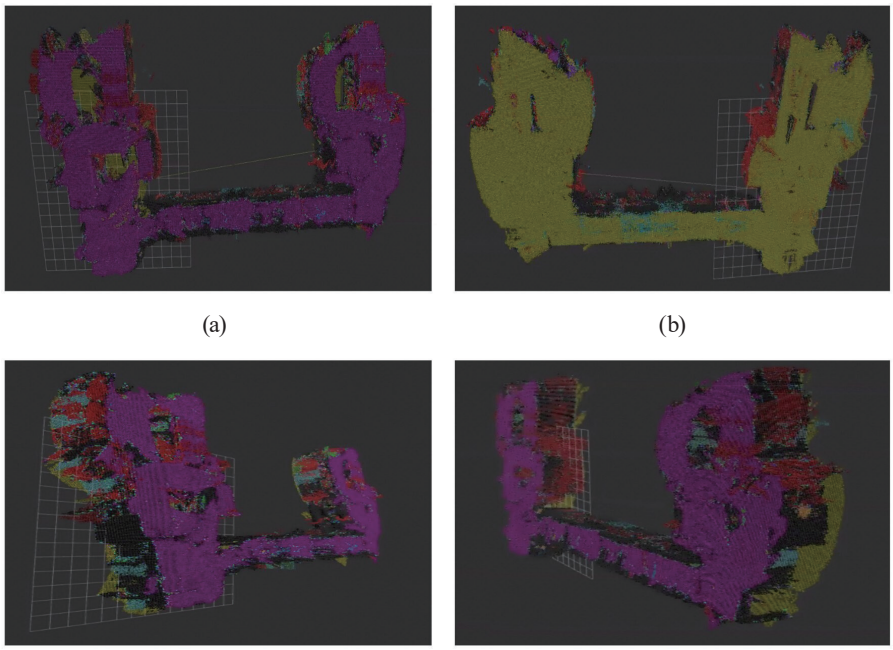
Camera.k1: −0.056191012263298035
Camera.k2: 0.06601350754499435
Camera.p1: 3.738758823601529e-05
Camera.p2: 0.0008224491029977798
Camera.k3: −0.021289801225066185

Camera.width: 1280
Camera.height: 720
Camera.fps: 30.0
Camera.RGB: 1
DepthMapFactor: 1000.0



(a) (b) (c)

Fig. 9. (Color online) Mapping process. Mapping (a) 0, (b) 50, and (c) 100%.



(a) (b)

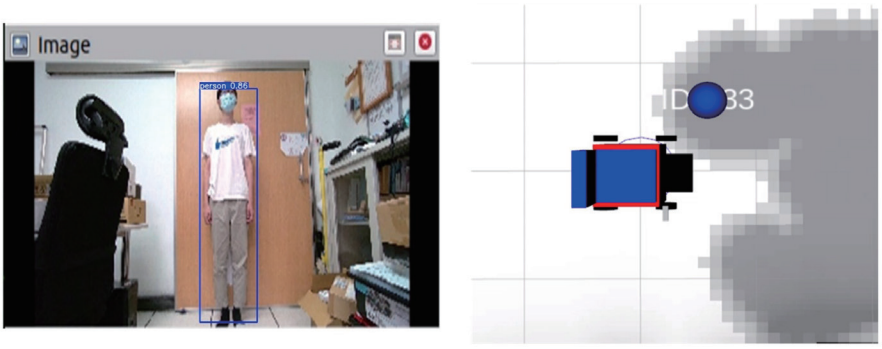
(c) (d)

Fig. 10. (Color online) 3D semantic map. (a) Top, (b) bottom, (c) left, and (d) right views.

Fig. 11. (Color online) 2D map.

image coordinates. It then samples and filters the corresponding depth information within the detected region to estimate the 3D spatial coordinates of the target. To reduce computational load and enhance real-time performance, only the center point of the bounding box is used as the representative obstacle point. This point is transformed into the base_link coordinate frame and packaged as an obstacle_msgs/ObstacleArray message, which is then published to the obstacles topic for use by the TEB local planner. Figure 12(a) shows the detection results in the camera view and Fig. 12(b) the corresponding visualization in RViz.

Subsequently, the costmap_converter node integrates the detected obstacles into the local_costmap, allowing the TEB Local Planner to consider real-time obstacle avoidance during trajectory planning. When a conflict is detected between the current path and a dynamic obstacle, TEB reoptimizes the trajectory to generate a feasible path, updates the /cmd_vel topic, and outputs the corresponding linear and angular velocity commands through the low-level



(a) (b)

Fig. 12. (Color online) Obstacle release diagram. (a) Camera detection and (b) RViz screens.

controller. These commands are issued at a frequency of 20 Hz to drive the smart wheelchair, enabling it to react in real time to moving pedestrians or unstructured obstacles.

After completing the semantic mapping, we integrated the navigation module to verify the system's autonomous planning and mobility capabilities in real-world environments. The navigation process is based on the 2D geometric map generated by ORB-SLAM, utilizing the Dijkstra algorithm for global path planning, whereas TEB handles real-time path adjustment and obstacle avoidance. Figure 13 presents the overall mission map, marking both the starting and goal positions, along with the initial global path. Figures 14 and 15 illustrate the wheelchair's position changes at different time points during navigation, shown from first-person and third-person perspectives, respectively. Subfigure (a) depicts the starting state, whereas (o) represents the moment of arrival at the goal. Subfigures (d) and (j) demonstrate the system's ability to successfully avoid dynamic obstacles during the process. The results confirm that the proposed system can reliably perform navigation and obstacle avoidance tasks in dynamic indoor environments, achieving the research objective of autonomous guidance for smart wheelchairs.

3.3 RelTR

In this paper, the RelTR model adopts ResNet-50 as the backbone feature extraction network, combined with a two-stage Transformer Decoder architecture to predict visual relationship triplets. Table 3 provides a detailed specification of each module layer, including convolutional kernel size (k), the number of channels (c), stride (s), repetition count (r), and output feature map dimensions, offering a comprehensive overview of the network structure used for model training. To support the real-world navigation process of the smart wheelchair, we designed an integrated trigger mechanism for image acquisition and cross-platform transmission, enabling the automatic execution of VRD upon navigation completion. Within the ROS Noetic environment, a ROS node named auto_capture_node was implemented to subscribe to the status information of the move_base module and monitor for the "GOAL Reached!" message indicating successful navigation. Once the target is reached, the system captures the current frame via a camera and automatically saves the image sequentially with numeric filenames (e.g., 001.jpg and 002.jpg) in a predefined directory named vrd_send_pic/pictures.

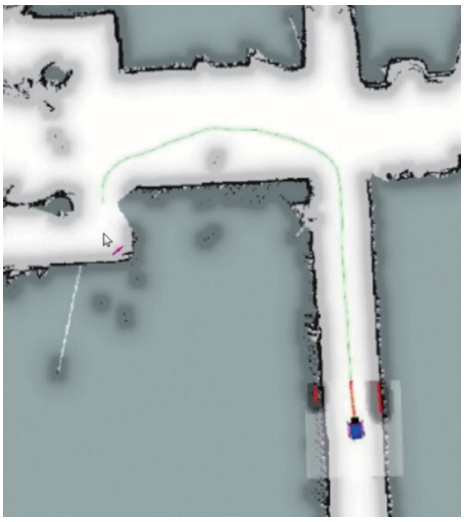


Fig. 13. (Color online) Mission map (green line is Dijkstra, red line is TEB).

Fig. 14. (Color online) Navigation process (wheelchair perspective).

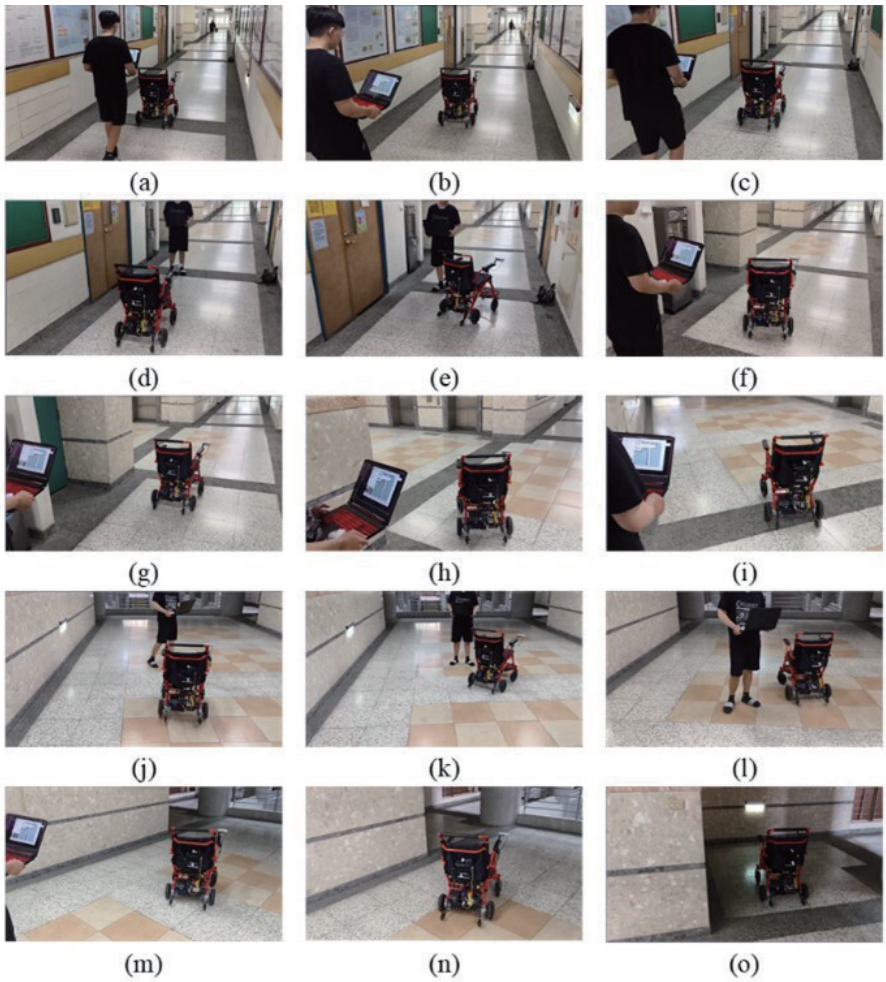


Fig. 15. (Color online) Navigation process (third perspective).

Table 3
RelTR training network architecture table.

Stage	Structure definition				Output size
	k	*c*	*s*	*r*	
Input	—	3	—	—	(H, W, 3)
Conv1	7 × 7	64	2	1	(H/2, W/2, 64)
MaxPool	3 × 3		2	1	(H/4, W/4, 64)
Conv2_x	3 × 3	256	1	3	(H/4, W/4, 256)
Conv3_x	3 × 3	512	2	4	(H/8, W/8, 512)
Conv4_x	3 × 3	1024	2	6	(H/16, W/16, 1024)
Conv5_x	3 × 3	2048	2	3	(H/32, W/32, 2048)
Conv+Flatten	1 × 1	256	2	1	(HW/1024, 256)
Position encoding	—	256	1	1	(HW/1024, 256)
Transformer encoder	—	256	—	6	(HW/1024, 256)
Transformer decoder	—	256	—	6	(100, 256)
Transformer decoder	—	256	—	6	(200, 256)
Relationship classification	—	256→Rel_class	—	1	(200, #rel_cls)
Bounding box regression	—	256→4	—	1	(200, 4)

Considering that the system is deployed on both the NVIDIA Jetson Orin and a laptop platform, we further implemented an image transmission mechanism based on Python Socket to achieve data synchronization between the two ends. The sender first queries the IPv4 address of the laptop, establishes a TCP connection, and then transmits a four-byte integer representing the length of the JPEG image, followed by the image byte data. Upon reading the length, the receiver can fully receive and reconstruct the image, which is then stored in the `RelTR/picture` directory as input for subsequent relationship detection.

On the laptop side, an ACER Nitro 5 AN515-54 is used to perform relationship inference and visualization. The software environment is built with Python 3.6, PyTorch 1.6, and includes packages such as scipy, matplotlib, and pycocotools. After completing the environment setup and loading the trained weights, the system automatically selects the most recently named image (i.e., the one with the highest numerical filename) from the RelTR/picture directory for inference, as illustrated in Fig. 16.

During the relationship detection inference process, the model structurally predicts possible subject–predicate–object (S–P–O) triplets from the input image. It leverages the CSA, DVA, and DEA attention mechanisms to enhance the semantic alignment between the subject and the object, and to aggregate relevant regional features. The inference results include the Top-10 triplets with the highest confidence scores, along with the bounding boxes of the predicted subject and object, and corresponding attention heatmaps.

As shown in Fig. 17, the visualization output consists of three images: the subject attention distribution, the object attention distribution, and the original image overlaid with bounding boxes. These are rendered using `matplotlib` to improve the interpretability and visual clarity of the results. Finally, the system also provides voice feedback to announce the inferred relationships, assisting users in understanding the semantic relationships of the surrounding objects in real time.

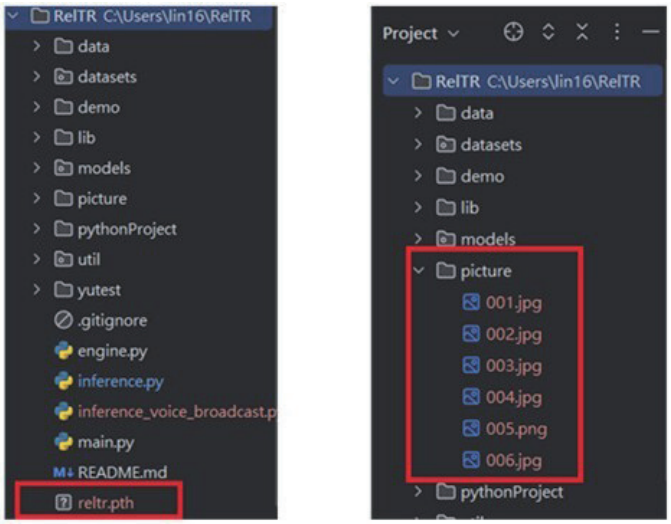


Fig. 16. (Color online) Data catalog.

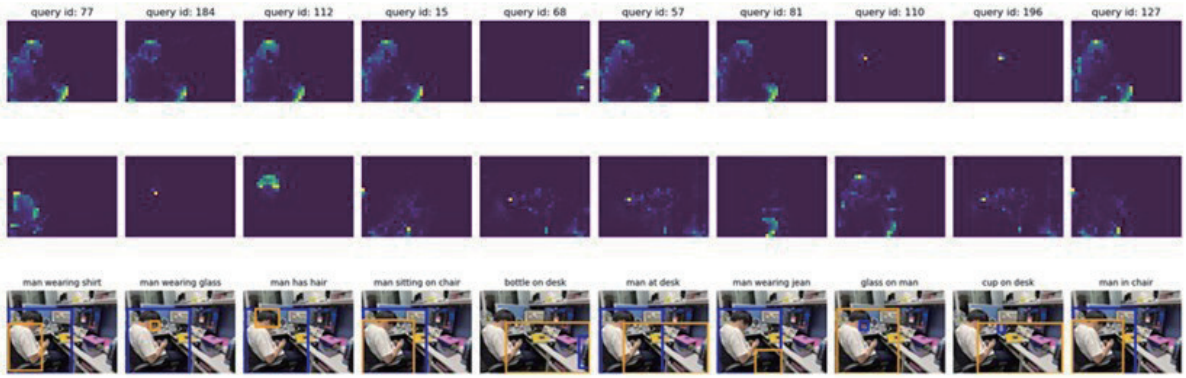


Fig. 17. (Color online) RelTR result diagram.

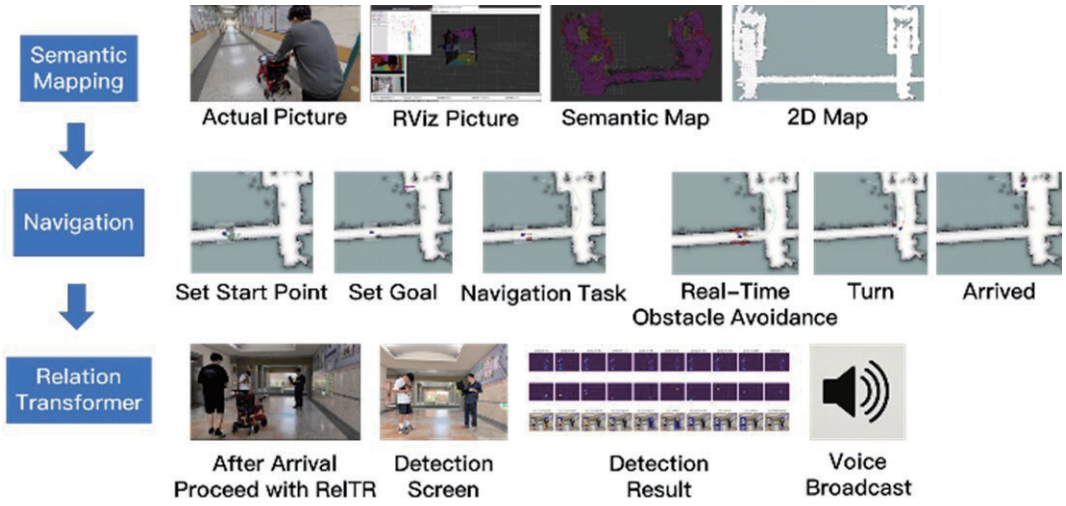


Fig. 18. (Color online) Smart wheelchair system integration diagram.

3.4 Integration results

To validate the successful integration and stable operation of the proposed modules—semantic mapping, autonomous navigation, and VRD—, we designed a demonstration task simulating the complete operational scenario of the smart wheelchair in a real-world environment, as described in this section. The task flow includes the following: the user specifies a target location via the interface and the system then automatically performs path planning and autonomous navigation. Upon reaching the destination, the VRD module is triggered to analyze the semantic context of the surrounding environment. Finally, the results are broadcast via a voice interface to help the user understand the semantic relationships among objects in the current space. The complete process is illustrated in Fig. 18.

4. Conclusion

In this study, we successfully implemented an intelligent wheelchair system that integrates visual semantic understanding, environmental mapping, and autonomous navigation, aiming to assist users in safely and intelligently completing mobility and interaction tasks within indoor

environments. The system incorporates semantic SLAM technology by utilizing an RGB-D camera sensor and combining the real-time localization and mapping capabilities of ORB-SLAM with the scene comprehension power of semantic segmentation models. This enables the real-time construction of semantically annotated 3D maps, which are efficiently compressed using Octomap to improve computational and memory efficiency. For navigation, the system employs a 2D geometric map for global path planning and uses the TEB local planner to generate dynamically adaptable and smooth paths, accounting for wheelchair kinematics and environmental changes to enhance flexibility and safety. In terms of scene understanding, the RelTR model is introduced to infer semantic triplets of object relationships from visual input. The results are then conveyed to users through voice feedback, enhancing their spatial awareness and decision-making ability.

Experimental results demonstrated that the system can reliably perform localization and mapping, while also achieving effective performance in navigation and semantic reasoning tasks. Ultimately, the system completes a full operation flow—from scene construction and autonomous navigation to relationship recognition and voice output—highlighting its high level of integration and strong potential for real-world applications.

References

1 World report on ageing and health: https://www.who.int/publications/i/item/9789241565042 (accessed August 2024).
2 Ministry of the Interior, R.O.C. (Taiwan): https://www.moi.gov.tw/News_Content.aspx?n=9&s=324767 (accessed August 2024).
3 Ministry of Health and Welfare: https://dep.mohw.gov.tw/dos/cp-5223-62358-113.html (accessed August 2024).
4 Y.-C. Lin, M.-H. Chuang, and Y.-M. Fang: Primary Care Med. **22** (2007) 235. https://doi.org/10.6965/PMCFM.200707.0235
5 R. Smith, M. Self, and P. Cheeseman: Proc. 1987 IEEE Int. Conf. Robotics and Automation (IEEE, 1987) 267–288. https://doi.org/10.1109/ROBOT.1987.1087846
6 M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit: Proc. 18th National Conf. Artificial Intelligence (AAAI, 2002) 593–598. https://dl.acm.org/doi/10.5555/777092.777184
7 R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós: IEEE Trans. Rob. **31** (2015) 1147. https://doi.org/10.1109/TRO.2015.2463671
8 GitHub: https://github.com/floatlazer/semantic_slam (accessed August 2024).
9 P. E. Hart, N. J. Nilsson, and B. Raphael: IEEE Trans. Syst. Sci. Cybern. **4** (1968) 100. https://doi.org/10.1109/TSSC.1968.300136
10 A. Stentz: Proc. 1994 IEEE Int. Conf. Robotics and Automation (IEEE, 1994) 3310–3317. https://doi.org/10.1109/ROBOT.1994.351061
11 S. Koenig and M. Likhachev: Proc. 18th National Conf. Artificial Intelligence (AAAI, 2002) 476–483. https://dl.acm.org/doi/10.5555/777092.777167
12 D. Fox, W. Burgard, and S. Thrun: IEEE Rob. Autom. Mag. **4** (1997) 23. https://doi.org/10.1109/100.580977
13 C. Rösmann, F. Hoffmann, and T. Bertram: 2015 European Control Conf. (ECC) (IEEE, 2015) 3352–3357. https://doi.org/10.1109/ECC.2015.7331052
14 Y. Cong, M. Y. Yang, and B. Rosenhahn: IEEE Trans. Pattern Anal. Mach. Intell. **45** (2023) 11169. https://doi.org/10.1109/TPAMI.2023.3268066
15 N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko: ECCV 2020: 16th European Conf. (ECCV, 2020) 213. https://doi.org/10.1007/978-3-030-58452-8_13
16 Merits Health: https://www.meritshealth.com.tw/power-wheelchairs/p113 (accessed August 2024).
17 H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia: Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR, 2017) 6230–6239. https://doi.org/10.48550/arXiv.1612.01105
18 A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard: Auton. Rob. **34** (2013) 189. http://doi.org/10.1007/s10514-012-9321-0
19 S. Thrun: Commun. ACM **45** (2002) 52. https://doi.org/10.1145/504729.504754
20 GitHub: https://github.com/ultralytics/yolov5 (accessed August 2024).