Fast Navigation of a Fuzzy-controlled Wheeled Robot Through the Combination of Expert Knowledge and Data-driven Multiobjective Evolutionary Learning

Chia-Feng Juang, *Fellow*, *IEEE*, Ching-Yu Chou, and Chin-Teng Lin, *Fellow*, *IEEE*

***Abstract*—This paper proposes a fast navigation scheme for a wheeled robot in unknown environments. The navigation scheme consists of obstacle boundary following (OBF), target seeking (TS), and vertex point seeking (VPS) behaviors and a behavior supervisor. The OBF behavior is achieved by a fuzzy controller (FC). This paper formulates the FC design problem as a new constrained multiobjective optimization problem and finds a set of nondominated FC solutions through the combination of expert knowledge and data-driven multiobjective ant colony optimization. The TS behavior is achieved by new fuzzy proportional-integral-derivative (PID) and proportional-derivative (PD) controllers that control the orientation and speed of the robot, respectively. The VPS behavior is proposed to shorten the navigation route by controlling the robot to move toward a new subgoal determined from the vertex point of an obstacle. A new behavior supervisor that manages the switching among the OBF, TS, and VPS behaviors in unknown environments is proposed. In the navigation of a real robot, a new robot localization method through the fusion of encoders and an infrared localization sensor using a particle filter is proposed. Finally, this paper presents simulations and experiments to verify the feasibility and advantages of the fast navigation scheme.**

***Keywords: Robot navigation, obstacle avoidance, multiobjective optimization algorithms, evolutionary fuzzy control, data-driven fuzzy systems*.**

I. INTRODUCTION

Automatic navigation of a wheeled robot in an environment is an important task in robot applications. The navigation of a robot can be performed in an environment with a known map or in an unknown environment, where the latter is a much more challenging task than the former. In an unknown environment, the entire map of the environment is not given in advance and a robot must online determine its local path based on its sensor measurements. This paper considers navigation of a robot in unknown environments. One important task in robot navigation is obstacle avoidance. To complete this task, fuzzy control of mobile robots has been widely studied [1]-[15].

Manuscript received April, 2020. This work was supported by the Ministry of Science and Technology, Taiwan, under Grant MOST 106-2221-E-005-004-MY2 and MOST 108-2638-E-005-001-MY2.

Chia-Feng Juang and Ching-Yu Chou are with the Department of Electrical Engineering, National Chung Hsing University, Taichung 402, Taiwan, R.O.C. (e-mail: cfjuang@ dragon.nchu.edu.tw; krad05181012@gmail.com). Chin-Teng Lin is with the CIBCI Lab, Centre for AI, FEIT, University of Technology Sydney, New South Wales, Australia (e-mail: Chin-Teng.Lin@uts.edu.au).

To automate the design of FCs, learning of FCs through neural learning [1], [12] or evolutionary computation algorithms [2]-[7], [14], [15] for obstacle avoidance has been proposed. For the later, various single-objective evolutionary computation algorithms have been proposed [2]-[6], [11]. The FCs in these algorithms consider only single-objective optimization, such as minimization of the distance error between a robot and an obstacle. Multiobjective ant colony optimization (ACO) of a set of FCs that consider the trade-off between robot-obstacle distance and the robot's moving speed has been proposed to control a robot to perform obstacle boundary following (OBF) [7], [14], [15]. These studies use the same objective functions in evaluating the OBF performance and do not consider the navigation problem. In addition to learning algorithms, the OBF performance depends heavily on the definitions of objective functions. In contrast to these studies, this paper formulates the robot OBF task as a new constrained multiobjective optimization problem to improve the OBF performance.

Another important task in navigating a robot is target seeking (TS). Orientation and speed control of a TS robot using proportional-integral-derivative (PID) [16], [17] or scheduled PID [18] controllers has been proposed posed is a typical approach in TS. In addition, the use of Mamdani-type fuzzy rules to determine PID coefficients for robot locomotion control has been proposed in [9], [13], [19], [20], where tens of manually designed fuzzy rules were used. The determination of PID coefficients using 25 type-2 fuzzy rules for robot orientation control was proposed in [21]. These methods need a large rule base, which increases the computational load, and the determination of the large number of rule parameters is a cumbersome task. For this problem, this paper proposes Takagi-Sugeno-Kang (TSK)-type fuzzy PID and proportional--derivative (PD) controllers to simplify the controller design task. Only two fuzzy rules with a single input variable are designed in each fuzzy PID/PD controller set, which significantly simplifies the design task.

For robot navigation in known environments, several global path plaining approaches have been proposed to find an optimal path in a given map [22]-[36]. Among them, one popular approach is grid-based planning algorithms, such as Dijkstra algorithm [22] and A*-based algorithms [23]-[25]. This approach runs on a grid-based structure with planning graphs given by grid vertices and edges, which constrains the headings of the paths. To release this constraint, theta* algorithm [26] and sampling-based planning (SBP) approach

[27] have been proposed. The SBP approach performs randomized sampling in search space, where commonly used algorithms include probabilistic roadmap method [28], Rapidly-exploring Random Trees (RRT) [29], and variants of RRT such as RRT* [30], [31]. Another path-planning approach is evolutionary computation algorithms [32], [33], which finds an optimal path through iterations of evolutions. The above global path planning approaches are computationally expensive and are most suitable for planning a path on a given map.

For navigation in unknown environments, the robot must online determine its local path based on its sensor measurements. Various navigation approaches in unknown environments have been proposed. One popular approach is the potential field approach [34], [35]. This approach shows the inherent limitations of possible oscillations in the presence of closely spaced obstacles or narrow passages, and the robot may be trapped in a dead cycle [34]. Online path planning using the dynamic window approach (DWA) that finds an optimal path in a valid local search space considering robot dynamics and collision avoidance has also been proposed [36], [37]. To apply the global path planning approaches to unknown environments, construction of a local map using the simultaneous localization and mapping (SLAM) followed by a path planning algorithm such as the A* algorithm in the map has been proposed [38], [39]. To improve the navigation path, the incorporation of DWA into SLAM-based A* [40], [41] or RTT* [42] algorithm has been proposed. A package of the SLAM-based DWA-A* algorithm is also available inside the ROS navigation stack [43]. Another common approach is the straightforward switching rule of performing either TS or bypassing an obstacle. This is a map-free approach and different map-free methods without [4], [7], [8]-[11] or with [1], [2], [18] the consideration of the dead-cycle problem have been proposed. However, these methods do not consider navigating the robot through a shorter path. In contrast to these methods, this paper proposes a map-free fast navigation scheme in which the robot can online determine a shorter path to the target without being trapped in a dead cycle.

In real navigation of a robot, localization is an essential task regardless of whether the environment is known or unknown. Localization using a laser range finder has been proposed in many studies based on the SLAM algorithm [38]-[44]. This approach is computationally expensive and typically needs iterative computations in an environment to construct an accurate environment map from laser scanning to obtain accurate localization. The StarGazer sensor that localizes a moving object based on markers on a ceiling and infrared light emitted by the robot has been developed (http://www.hagisonic.com). Determining the location of a robot using the StarGazer sensor has been proposed [18], [45]. This paper proposes the fusion of the StarGazer sensor and encoders using the particle filter to correct the Stargazer sensor errors, especially in the overlapping areas covered by different markers.

The contributions of this paper are threefold. First, this paper proposes new objective functions and constraints to improve the control performance of an OBF robot. To improve the learning performance of the FCs designed under this new constrained optimization formulation, this paper also proposes the combination of data-driven multiobjective ACO with expert knowledge expressed by fuzzy rules, which takes advantage of the interpretable property of a fuzzy control rule. Second, this paper proposes a fast map-free navigation scheme consisting of the fuzzy OBF and fuzzy PID/PD TS behaviors with a new vertex point seeking (VPS) behavior proposed to shorten the navigation path. Finally, the proposed fast navigation scheme based on fuzzy control of the robot is applied to navigate a real robot with a new localization approach. Experimental results show the efficiency of the navigation scheme and high control accuracy when the robot reaches a target.

This paper is organized as follows. Section II introduces the rules in the FCs, learning of FCs in executing the OBF behavior using a newly formulated constrained multiobjective optimization problem, and the multiobjective ACO. Section III introduces the speed and orientation control by fuzzy PID and PD controllers during the TS behavior. Section IV introduces the operation of the VPS behavior and the proposed fast navigation scheme. Section V introduces the localization of a real robot. Section VI and Section VII present simulations and experimental results, respectively. Finally, Section VIII summarizes conclusions.

II. MULTIOBJECTIVE FUZZY CONTROL OF AN OBSTACLE BOUNDARY FOLLOWING ROBOT

This section introduces the FC, the wheeled mobile robot, fuzzy control of the robot for OBF, and its multiobjective learning configuration using the multiobjective front-guided continuous ACO (MO-FCACO) [7].

A. Robot Fuzzy Control

This paper applies a zero-order TS-type FC to control the robot to execute the OBF behavior. For the fuzzy set A_j^i in input variable $x_j(t)$ of rule i, the following Gaussian membership function is used:

$$\mu_j^i(x_j) = \exp\left\{-\left((x_j - m_j^i)^2 / (\sigma_j^i)^2\right)^2\right\} \tag{1}$$

where m_j^i and σ_j^i represent the center and width, respectively. By using the product-based AND operation and the weighted average defuzzification, the p th output y_p of the FC is

$$y_p = \frac{\sum_{i=1}^{r}\prod_{j=1}^{n}\mu_j^i(x_j)\cdot a_p^i}{\sum_{i=1}^{r}\prod_{j=1}^{n}\mu_j^i(x_j)} \tag{2}$$

where a_p^i is the consequent value of y_p in rule i and r is the number of fuzzy rules.

The controlled robot is the PIONEER 3-DX (http://robots.mobilerobots.com/) with a laser SICK LMS-100 installed, as shown in Fig. 1. The maximum sensor measurement fed to the FC is set to 1 m, with 1 degree of

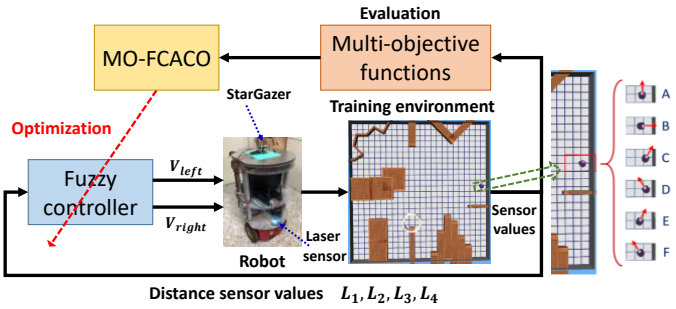


Fig. 1. The robot and the data-driven multiobjective learning configuration of FCs for an OBF robot starting from six different initial poses.

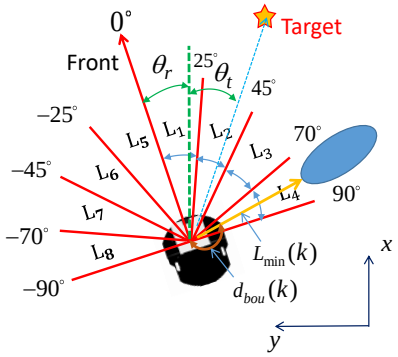


Fig. 2. Four sensor ranges on each side of the robot, the minimum sensor measurement $L_{\min}(k)$, and the angle deviation between the directions of the robot and the target.

angular resolution. For the right-side OBF, the right half of the sensor range of laser measurements is divided into four ranges, as shown in Fig. 2. The minimum sensor value $L_i(k)$ of range i is fed as the FC input. The FC outputs two velocities $v_{left}, v_{right} \in [-15.6, 15.6]$ rad/s (1 rad/s= 0.098 m/s) to the left and right wheels. In addition to the four laser sensor measurements, the difference $\Delta v(k-1) = v_{left}(k-1) - v_{right}(k-1)$ that controls the robot at the last time step is also fed as the FC input. Each fuzzy rule in the right OBF FC is described by

$$R^i_{right}: \text{If } L_1(k) \text{ is } A^i_1 \text{ and } L_2(k) \text{ is } A^i_2 \text{ and } L_3(k) \text{ is } A^i_3 \text{ and } L_4(k) \text{ is } A^i_4 \text{ and } \Delta v(k-1) \text{ is } A^i_5 \text{ then } v_{left}(k) \text{ is } a^i_1 \text{ and } v_{right}(k) \text{ is } a^i_2 \quad (3)$$

The FC parameters are all learned through a data-driven approach introduced in Section II-B. In addition to learning from scratch, one advantage of using the FC instead of other nonlinear controllers is that *a priori* expert knowledge can be expressed in the form of fuzzy if-then rules and incorporated into the FC before learning. The incorporation of these control rules helps improve learning performance. To take advantage of this inherent property, this paper proposes a design that combines expert knowledge and a data-driven MO-FCACO learning approach. This paper incorporates the following two human knowledge-based fuzzy control rules in the FC for the OBF problem:

$$R^{right}_1: \text{If } L_1(k) \text{ is Short and } L_4(k) \text{ is Short then } v_{left}(k) \text{ is } -\omega \text{ and } v_{right}(k) \text{ is } \omega \quad (4)$$

$$R^{right}_2: \text{If } L_1(k) \text{ is Far and } L_4(k) \text{ is Far Then } v_{left}(k) \text{ is } \omega \text{ and } v_{right}(k) \text{ is } -\omega \quad (5)$$

The first and second control rules are assigned to control the robot around inner and outer corners, respectively. When the robot is around an inner corner, the distance values of $L_1(k)$ and $L_4(k)$ are small, which accounts for the design of the antecedent part in the first rule. In this condition, the robot should make a left turn, which is accomplished by rotating the right and left wheels forward and backward, respectively, as inferred from the consequent part of the first rule. Similarly, when the robot is around an outer corner, the distance values of $L_1(k)$ and $L_4(k)$ are large, which accounts for the design of the antecedent part in the second rule. In this condition, the robot should make a right turn, as inferred from the consequent part of the second rule. The initial centers of the fuzzy sets "Short" and "Far" are set to 0.3 m and 0.7 m, respectively. The initial consequent parameter ω is set to 5 rad/s. Similar to the rules in (3), all the initial parameters in (4) and (5) are also optimized by using the MO-FCACO algorithm. Based on the left-right symmetry, the learned FC for the right OBF can be directly modified to obtain the FC for the left OBF without retraining.

B. Multiobjective Learning Configuration for Obstacle Boundary Following

The evolutionary MO-FCACO algorithm is used to endow the robot with the capability of learning multiobjective FCs by itself. This self-learning technique is designed to resolve the OBF problem that considers two conflicting objectives: maintaining a proper robot-obstacle distance and moving at a high speed. To address this problem, this paper applies a multiobjective evolutionary learning configuration to find a set of FCs showing trade-offs between the two objectives. Fig. 1 shows the self-learning configuration of the OBF robot. The FC is learned through consecutive control trials and evolutionary parameter optimization to accomplish the OBF task. The training environment contains diverse concave and convex boundaries, as shown in Fig. 1. Starting from point "A" in the training process, the mobile robot should successfully execute the OBF behavior with T_{total} (=2000) time steps. In addition, starting from another five different initial poses (positions "B" to "F" in Fig. 1), the robot is required to successfully execute the OBF behavior for 30 time steps, after which the robot moves along the straight boundary.

The preparatory work for evolutionary learning is the formulation of the control problem as a constrained multiobjective optimization problem. In contrast to previous papers on multiobjective evolutionary OBF robots [7], [14], [15] this paper proposes the following new objective functions and constraints to improve the OBF performance.

Objective 1 (f_1): The first control objective is that the robot should maintain a constant distance from the followed obstacle boundary. The objective function is defined as

$$f_1 = \frac{\sum_{k=1}^{T_c} \left| \frac{L_{\min}(k)}{d_{wall} + d_{bou}(k)} - 1 \right| + 100\delta(T_c)}{T_c} \quad (6)$$

where T_C is the time step at which the robot stops. The constant d_{wall} (=0.15 m) is the desired distance between an obstacle boundary and its nearest robot boundary in the right-side range $[0^\circ, 90^\circ]$. The measurement $L_{\min}(k)$ is the minimum sensor measurement in the right-side range, as shown in Fig. 2. The distance $d_{bou}(k)$ is the physical distance (the maximum is $d_{bou-\max}$ =0.2) between the laser center and the robot boundary in the direction of $L_{\min}(k)$, as shown in Fig. 2. The term $\delta(\cdot)$ in (6) is a penalty value with an initial value of $\delta(0)$ =0 and is defined as follows:

$$\delta(k+1) = \begin{cases} \delta(k)+1 \text{ , } C(k) \geq C_{th} \\ \delta(k), \quad \text{otherwise} \end{cases} \quad (7)$$

where $C(k)$ is a counter defined as follows:

$$C(k+1) = \begin{cases} C(k)+1, \ |L_{\min}(k) - d_{bou}(k)| > d_{wall} + d_{bou-\max} \\ C(k)-1, \ |L_{\min}(k) - d_{bou}(k)| \leq d_{wall} + d_{bou-\max} \\ \qquad \text{and } C(k) > 0 \\ 0, \quad \text{otherwise} \end{cases} . \quad (8)$$

In contrast to the objective function defined in [7], [14], [15], the new term $\delta(k)$ is included to avoid a large deviation from the desired robot-obstacle distance, especially when the robot moves along outer corners. If the robot is not close to the wall (condition 1 in (8)), the counter $C(k)$ will start being increased by one at each time step. The penalty term $\delta(k)$ increases when the robot is not close to the wall for more than C_{th} successive time steps, i.e., $C(k) \geq C_{th}$ in (7). The value of C_{th} is set to 10 because the robot typically takes around 10 time steps to move around a corner.

Objective 2 (f_2): The second control objective is that the robot should move at high speed. The objective function is defined as follows:

$$f_2 = \frac{1}{\sum_{k=1}^{T_c} \left| \frac{V_{speed}(k)}{T_c} \right|} \quad (9)$$

where V_{speed} is the moving speed of the robot.

In the learning process, an FC is deemed to have failed if one of the following four constraints is violated. These constraints are imposed in the learning process to avoid the convergence to failed FCs. When an FC fails, the robot stops, and the time step is recorded as T_C, as described above. The first constraint is that the robot should not move far away (greater than distance $W_f(k)$) from the obstacle boundary for a long duration (set to 2 C_{th} =20). This constraint is formulated as follows:

$$F_e(k) \leq 20 \quad (10)$$

where

$$F_e(k) = \begin{cases} F_e(k-1)+1, \ L_{\min}(k) > W_f(k) \\ 0, \ \text{otherwise} \end{cases} \quad (11)$$

and

$$W_f(k) = \begin{cases} 10 \cdot d_{wall}, & k < \frac{1}{4} T_{total} \\ 2 - \frac{2k}{T_{total}}, & \frac{1}{4} T_{total} \leq k \leq \frac{3}{4} T_{total} \\ d_{bou-\max} + 2d_{wall}, & k > \frac{3}{4} T_{total} \end{cases} . \quad (12)$$

The tolerable distance $W_f(k)$ decreases with time as shown in (12). The motivation is that a looser distance constraint is assigned at the initial control time period to allow the robot to learn more flexibly in the early learning stage. A more conservative distance constraint is assigned to the robot during the final control time period to quickly distinguish the performance of different FCs.

The second constraint is that the robot cannot too close to the obstacle, i.e., $L_{\min}(k) - d_{bou}(k) > 0.1\ m$. The third constraint is that the robot cannot move too slowly. The final constraint is that the robot cannot move backward.

The two objective functions in (6) and (9) together with the four constrains form a constrained multiobjective optimization problem. To solve this problem, a penalty value T_p is defined when one of the four constraints is violated; i.e., an FC fails. The penalty value is defined as $T_p = T_{total} - T_C$, which is the total number of uncompleted time steps. For a successful FC, T_p is equal to zero. Based on the definition of T_p, the constrained multiobjective optimization problem is reformulated as follows:

$$\hat{f}_i = f_i + T_P, \ i = 1, 2. \quad (13)$$

This paper sets the robot-obstacle distance d_{wall} in (6) to 0.15 *m*, a shorter distance than the value of 0.5 *m* in [7], [14], [15], [18], which induces a more challenging learning task. The learning methods in these previous studies fail when they are applied to control the robot with this stricter distance constraint. The FCs designed under the new mathematical formulation and the expert knowledge-leveraged data-driven approach show the advantage of controlling the robot in moving along obstacle boundaries with a smaller robot-obstacle clearance, which reduces traveling path distance and enables the robot to pass through narrow passages. When the assigned d_{wall} value is

smaller than 0.15 m, the robot may fail to perform the OBF behavior due to different uncertainties. Therefore, this paper conservatively sets the distance d_{wall} to 0.15 *m*.

C. *Multiobjective Front-guided Continuous Ant Colony Optimization (MO-FCACO)*

This section briefly describes the MO-FCACO algorithm [7]. The MO-FCACO evolves with *N* solutions $\vec{S}_1,\ldots,\vec{S}_N$ and the evolution operation consists of three phases. In the first phase, $2L$ temporary solutions $\hat{\vec{S}}_1,\ldots,\hat{\vec{S}}_l,\ldots,\hat{\vec{S}}_{2L}$ are generated using elite and tournament selection schemes based on the idea of pheromone levels.

The second phase applies the Gaussian resampling operation to each solution component $\hat{S}_i^j$ at the $2L$ temporary solutions to mimic the ant wondering behavior along a path. The mean of the Gaussian probability density function (PDF) applied to the solution component $\hat{S}_i^j$ is set to $\hat{S}_i^j$. After the Gaussian resampling operation, the temporary solution vectors generated from $\hat{\vec{S}}_1,\ldots,\hat{\vec{S}}_{2L}$ are expressed as $\tilde{\vec{S}}_1,\ldots,\tilde{\vec{S}}_{2L}$.

In the final phase, each temporary solution $\tilde{\vec{S}}_l$ independently tracks a randomly and uniformly selected non-dominated solution $\vec{S}_{non}$. The final $2L$ new solutions are given as follows:

$$\vec{S}_{N+l} = \tilde{\vec{S}}_l + \vec{\phi} \otimes (\vec{S}_{non} - \tilde{\vec{S}}_l),\ l = 1,\ldots,2L \tag{14}$$

where $\vec{\phi}$ is a random vector whose entries are uniformly distributed random variables in [0, 1].

The original *N* solutions and the $2L$ new solutions are sorted and only the top *N* solutions at the end of the iteration are reserved.

III. TARGET SEEKING AND FUZZY CONTROL

This section introduces the TS behavior, which is achieved by controlling the robot orientation and speed using fuzzy PID and PD controllers, respectively. Details about controllers are introduced as follows.

A. Robot Orientation Control

Fig. 2 shows the angle deviation θ_{TS}, computed as follows:

$$\theta_{TS} = \theta_t - \theta_r,\ \ \theta_{TS} \in [-180^\circ, 180^\circ], \tag{15}$$

where θ_r and θ_t denote the orientations of the robot and target, respectively. Depending on $|\theta_{TS}|$, two orientation control rules are assigned. For a large orientation deviation $|\theta_{TS}|$, the main control objective is reducing the response time so that the robot can quickly face the target. For a very small $|\theta_{TS}|$, the main control objective is reducing overshoot when the orientation deviation approaches zero. Based on this expert knowledge,

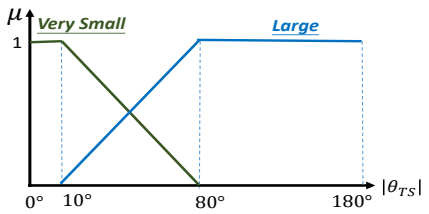


Fig. 3. Fuzzy sets for TS orientation control.

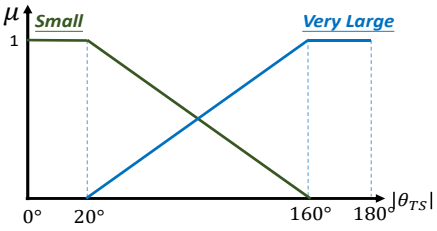


Fig. 4. Fuzzy sets for TS speed control.

this paper manages the orientation control by using the following two fuzzy control rules:

$$\begin{aligned}&\text{Rule 1: If } |\theta_{TS}(k)| \text{ is Large then } V_{ore}(k) \text{ is } v_{PID}^1(k)\\ &\text{Rule 2: If } |\theta_{TS}(k)| \text{ is Very Small then } V_{ore}(k) \text{ is } v_{PID}^2(k)\end{aligned} \tag{16}$$

where fuzzy sets "Large" and "Very Small" are described by trapezoidal membership functions, as shown in Fig. 3. The orientation control is achieved by applying the fuzzy PID controller to change the left and right wheel velocities as follows:

$$\begin{aligned}v_{right}(k) &= V_f(k) + 0.5 \times V_{ore}(k)\\ v_{left}(k) &= V_f(k) - 0.5 \times V_{ore}(k)\end{aligned} \tag{17}$$

where $V_f(k)$ is the forward moving speed of the robot and is determined by the fuzzy PD controller detailed at Section III.B. The consequent value $v_{PID}^i(k)$ is the output of a PID controller and is described as follows:

$$v_{PID}^i(k) = K_P^i \theta_{TS}(k) + K_I^i \sum_{t=1}^{k} \theta_{TS}(t)\Delta t + K_D^i \frac{[\theta_{TS}(k) - \theta_{TS}(k-1)]}{\Delta t} \tag{18}$$

where the units of $V_{ore}(t)$, θ_{TS}, and Δt are rad/s, degrees, and seconds, respectively. The coefficients are set to $K_P^1 = 0.2,\ K_I^1 = 0.05,$ and $K_D^1 = 0.02$ in rule 1 and $K_P^2 = K_P^1,\ K_I^1 = 0.01$, and $K_D^2 = K_D^1$ in rule 2, where a smaller K_I^1 coefficient is assigned in rule 2 to reduce the overshoot.

B. Robot Speed Control

This section introduces the fuzzy PD controller for the robot speed control. Depending on $|\theta_{TS}|$, two speed control rules are assigned. First, if the deviation angle $|\theta_{TS}|$ is very large, then the robot moves at a low speed v_{slow} = 4 rad/s for a better orientation control. If the deviation angle $|\theta_{TS}|$ is small, then the robot moves at a higher speed toward the target using a PD controller. Based on expert knowledge, this paper manages the speed control by using the following two fuzzy control rules:

$$\begin{aligned} &\text{If } |\theta_{TS}(k)| \text{ is Very Large then } V_f(k) \text{ is } v_{slow} \\ &\text{If } |\theta_{TS}(k)| \text{ is Small then } V_f(k) \text{ is } v_{PD}(k) \end{aligned} \tag{19}$$

where fuzzy sets "Very Large" and "Small" are described by trapezoidal membership functions, as shown in Fig. 4, and $v_{PD}(t)$ is the output of a PD controller described as follows:

$$v_{pd}(k) = K_P L_T(k) + K_D \frac{[L_T(k) - L_T(k-1)]}{\Delta t} \tag{20}$$

where $L_T(k)$ is the smaller of the distances between the robot and the target or an obstacle. That is, the fuzzy PD controller also controls the speed of the robot when it approaches an obstacle located between the robot and the target. The units of $L_T(k)$, and Δt are cm and seconds, respectively. The coefficients are set to K_P =10 and K_D =2. The final control outputs sent to the two robot wheels are determined by (17).

Finally, considering the moving inertia of a real robot, when the robot is approaching an obstacle or a target (i.e., $L_T(k) < 0.5\ m$), the moving speeds of the two wheels determined in (18) are reduced to

$$\begin{aligned} v_{right}(k) &= 0.5 \times V_f(k) + 0.5 \times V_{ore}(k) \\ v_{left}(k) &= 0.5 \times V_f(k) - 0.5 \times V_{ore}(k) \end{aligned} \tag{21}$$

The change reduces the forward moving speed so that the robot has enough time to reorient itself to face the correct direction when it is very close to the target/obstacle.

IV. FAST NAVIGATION IN UNKNOWN ENVIRONMENTS

The general navigation scheme consists of three parts: TS, OBF, and a behavior supervisor. To reduce the robot navigation time in unknown environments, this paper proposes a fast navigation scheme consisting of four parts, with the additional VPS behavior proposed to reduce the robot navigation time. The VPS behavior first finds convex vertex points, if any, of an obstacle when the robot is executing the OBF/TS behavior and then selects one as a subgoal to reach. The details of the VPS in TS and OBF behaviors and the behavior supervisor are introduced below.

A. Vertex Point Seeking During Target Seeking

While the robot is executing the TS behavior, the laser sensor helps sense the existence of an obstacle located far away in the direction of movement. Let (x_0, y_0) denote the 2D coordinates of the robot. The distance between the robot and the target is denoted by d_T. In the scanning region from -5° to 5°, as shown in Fig. 5, the maximum considered distance D_{VPS} is set to 6 m, which is sufficiently large in indoor environments. In this range, if the minimum sensor measurement is smaller than the robot-target distance, indicating an obstacle between the robot front and the target, then the robot starts to find vertex

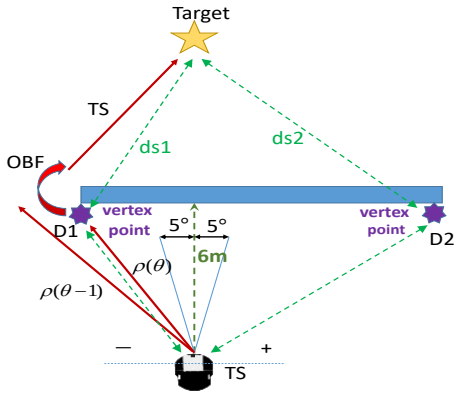


Fig. 5. Searching for two TS convex vertex points D1 and D2: as ds1< ds2, D1 is selected as the subgoal.

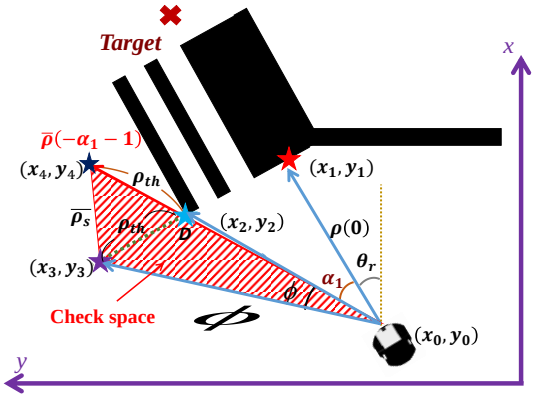


Fig. 6. The space defined to check if a convex vertex point is qualified as a subgoal candidate.

points in the obstacle. Starting from the robot's front direction at zero degrees, the laser searches for possible convex vertex points in both the clockwise and counterclockwise directions. Consider the counterclockwise laser scanning direction: if

$$\rho(\theta - 1) - \rho(\theta) > \rho_{th}, \quad \theta \in (-90^\circ, 0] \tag{22}$$

where $\rho(\theta)$ is the laser measurement at degree θ and ρ_{th} (= 1.5 m) is a threshold, then the coordinates of point D at which the distance $\rho(\theta)$ is measured are recorded as (x_2, y_2). The included angle between the robot's forward direction and the vertex point is denoted by α_1, as shown in Fig. 6. To determine whether point *D* can be selected as a subgoal candidate, the next step is checking if there is an enough space near this point for the robot to pass by the obstacle. Let (x_1, y_1) denote the coordinates of the obstacle scanned at laser distance $\rho(0)$ and (x_3, y_3) denote a point in the direction from (x_1, y_1) to (x_2, y_2) at distance ρ_{th} from (x_2, y_2), as shown in Fig. 6. The included angle between the directions from (x_0, y_0) to (x_2, y_2) and (x_3, y_3) is denoted by ϕ, as shown in Fig. 6. The distance between (x_0, y_0) and (x_3, y_3) is denoted by $\bar{\rho}(-\alpha_1 - \phi)$. To define the space area, at the laser sensor direction at the angle of

$-\alpha_1 - 1$, a new point (x_4, y_4) with the distance of $\rho(-\alpha_1) + \rho_{th}$ to (x_0, y_0) is defined. The area formed by (x_0, y_0), (x_3, y_3), and (x_4, y_4) should contain no obstacles, as shown in Fig. 6, which is confirmed by

$$\rho(-\alpha_1 - i) \geq \overline{\rho}_s(-\alpha_1 - i)\ , i = 1, ..., \phi \qquad (23)$$

where $\overline{\rho}_s(-\alpha_1 - i)$ is the distance from the laser to the space boundary at angle $-\alpha_1 - i$.

If both (22) and (23) are satisfied, then point *D* is regarded as a subgoal candidate, and the counterclockwise laser scanning stops. In the clockwise scanning direction, another subgoal candidate, if any, is found by the same approach. During the scanning process, if two subgoal candidates are found in the two scanning directions, such as *D*1 and *D*2 in Fig. 5, then the candidate with the minimum distance to the target (*D*1 in Fig. 5) is identified as a subgoal. Once a subgoal is found, the robot moves to the subgoal using the fuzzy PID/PD controller Described in Section III. The robot continues the original TS behavior at each time step if no subgoal is found.

B. Vertex Point Seeking During Obstacle Boundary Following

The robot begins the OBF convex vertex point scanning process once the distance measurement in the front direction is smaller than 4 m; this condition indicates the presence of an obstacle ahead. The search for a vertex point is constrained to only the boundary the robot is following to avoid the dead cycle problem of repeatedly switching between vertex points. To satisfy this constraint, the vertex scanning process starts only if

$$\rho(-45^o) < D_{meet}\ , \qquad (24)$$

where D_{meet} is the distance that the robot encounters an obstacle, as stated in Section IV-C. In addition, the separation distance *SD* between two neighboring laser scanning directions, as shown in Fig. 7, should satisfy

$$SD \approx (\rho(\theta + 1) - \rho(\theta)) \cos\theta > -d_{SD}\ . \qquad (25)$$

This inequality is imposed because *SD* is generally a small value (d_{SD} is conservatively set to 0. 5 m), except at the convex vertex point, for the same obstacle being followed. If (25) is not satisfied, then the vertex search process stops, and the robot continues the OBF behavior.

In the left OBF behavior, if (24) and (25) are satisfied, a possible OBF-vertex point is found in the right side, i.e., $\theta \in [0^\circ, 90^\circ)$. Similar to the VPS during TS, if

$$\rho(\theta + 1) - \rho(\theta) > \rho_{th}\ ,\ \theta \in [0^\circ, 90^\circ) \qquad (26)$$

then the search stops. The point on the obstacle boundary with the distance of $\rho(\theta)$ is identified as an OBF-vertex point as well as a subgoal, as shown in Fig. 7. The original OBF behavior is switched to the VPS behavior.

C. Behavior Supervisor

The behavior supervisor determines the switching among the TS, OBF, and VPS behaviors. Fig. 8 shows the flowchart of

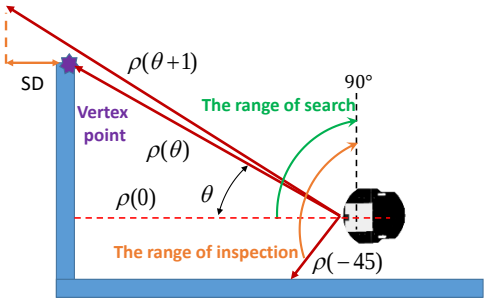


Fig. 7. Vertex point checking and searching ranges and the OBF-vertex point chosen as a subgoal while the robot is executing the OBF behavior.

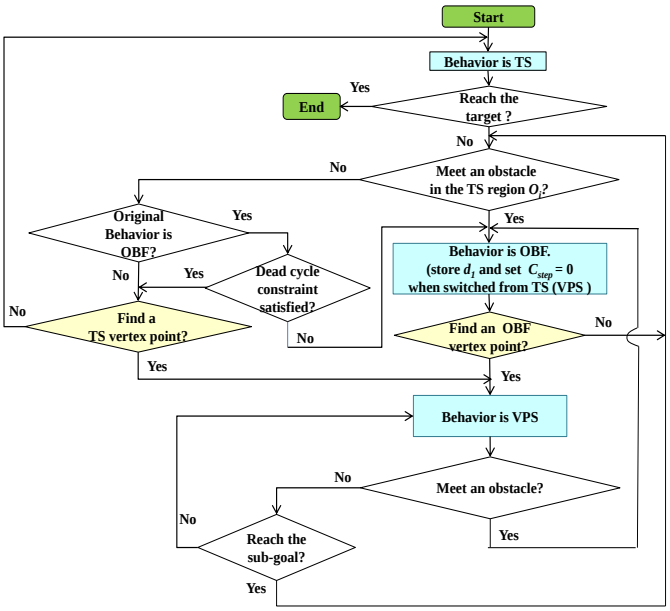


Fig. 8. Flowchart of the behavior supervisor with the VPS behavior in the fast navigation scheme.

the fast navigation scheme. The initial behavior of the robot is set to TS. The surroundings of the robot are divided into three overlapped regions: $O_1 = [-30^\circ, 30^\circ]$, $O_2 = [0^\circ, 180^\circ]$, and $O_3 = [0^\circ, -180^\circ]$. If the target is located in a region O_i in which the robot encounters an obstacle, i.e., the minimum distance sensor measurement is smaller than D_{meet} =1 *m*, then the robot switches to the OBF behavior. The avoidance of the dead-cycle problem is considered in the fast navigation scheme. The criteria of the robot-target distance and the time step counter, as proposed in [2], used in deciding to switch from the OBF to TS behavior are incorporated into the fast navigation scheme. In the proposed fast navigation scheme, when the supervisor switches the robot behavior from TS or TS-based VPS to OBF, the distance d_1 between the robot and the target is recorded, and the step counter c_{step} is set to zero. At the position where the robot does not meet an obstacle in the target region, the robot-target distance d_2 is determined, and counter c_{step} starts to be increased. If $d_2 < d_1$ or the robot senses that there is no obstacle between the robot and the target within the

sensing region, then after c_{step} (=5) steps of satisfying the constraint, the robot switches from the OBF to TS behavior.

V. ROBOT LOCALIZATION

This section introduces the proposed localization method for localizing the position and orientation of a real robot. The method consists of three parts: two wheel encoders, a StarGazer sensor (http://www.hagisonic.com) at the top of the robot, as shown in Fig. 1, and the particle filter algorithm. The StarGazer sensor localizes a robot based on markers on a ceiling and infrared light emitted by the robot.

The rotary encoder feeds back both the changes in the 2D position, $\Delta x(k)$ and $\Delta y(k)$, and the orientation $\Delta\theta_r(k)$ of the robot with respect to its previous state at each control time step k. The StarGazer sensor functions as an observer that measures the state $\tilde{\boldsymbol{z}}(k) = (\tilde{x}(k), \tilde{y}(k), \tilde{\theta}_r(k))$ containing the position $(\tilde{x}(k), \tilde{y}(k))$ and orientation $\tilde{\theta}_r(k)$ of the robot. The particle filter fuses the information from the encoder and the StarGazer sensor to provide the estimated state $\boldsymbol{x}(k) = (x(k), y(k), \theta_r(k))$ of the robot. The 3D state $\hat{\boldsymbol{x}}^i(k) = (\hat{x}^i(k), \hat{y}^i(k), \hat{\theta}_r^i(k))$ of particle i contains the 2D position and orientation of the robot. Initially, a group of P_S (=50) particles are randomly generated around the initial observed state. In the Bayes prediction step, the next state $\hat{\boldsymbol{x}}^i(k)$ of particle i is predicted as follows:

$$\begin{aligned} \hat{x}^i(k) &= \hat{x}^i(k-1) + \Delta x(k) + G(0, \sigma_x^2) \\ \hat{y}^i(k) &= \hat{y}^i(k-1) + \Delta y(k) + G(0, \sigma_y^2),\ i = 1, ..., P_S \\ \hat{\theta}_r^i(k) &= \hat{\theta}_r^i(k-1) + \Delta\theta_r(k) + G(0, \sigma_\theta^2) \end{aligned} \quad (27)$$

where $G(0, \cdot)$ represents a normal distribution with zero mean, and σ_x (=5 cm), σ_y (= 5 cm), and σ_θ ($=\pi/36$ rad$=5^o$) are standard deviations.

In the Bayes update step, the posterior probability $p(\boldsymbol{x}(k) \mid \tilde{\boldsymbol{z}}(1:k))$ at time *k* is represented by the P_S particles $\hat{\boldsymbol{x}}^i(k)$ and their associated weights $w^i(k)$ computed as follows:

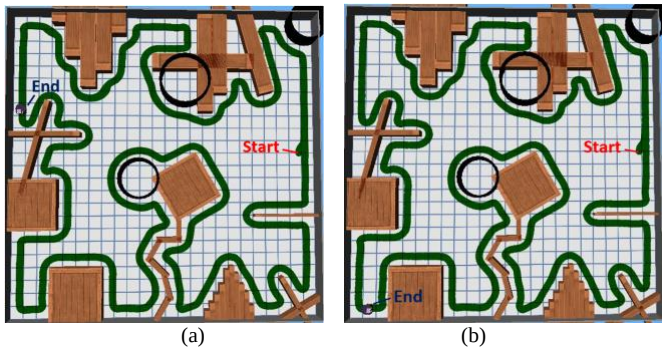
$$w^i(k) = \frac{1}{\hat{\sigma}_x \hat{\sigma}_y \hat{\sigma}_\theta (2\pi)^{3/2}} e^{-\frac{1}{2}\left\{\frac{(\hat{x}^i(k)-\tilde{x}(k))^2}{\hat{\sigma}_x^2} + \frac{(\hat{y}^i(k)-\tilde{y}(k))^2}{\hat{\sigma}_y^2} + \frac{(\hat{\theta}^i(k)-\tilde{\theta}^i(k))^2}{\hat{\sigma}_\theta^2}\right\}} \quad (28)$$

where the standard deviations are set to be $\hat{\sigma}_x = 2\sigma_x$, $\hat{\sigma}_y = 2\sigma_y$, and $\hat{\sigma}_\theta = 2\sigma_\theta$.

Given the weights $w^i(k)$, the next step performs P_S samplings of the particles based on their weights. The tournament selection technique is used in the resampling operation. In this selection, five particles are randomly and uniformly selected regardless of their weights, and the one with the highest weight is retained. Finally, the state $\boldsymbol{x}(k)$ is estimated from the average of the P_S selected particles.

TABLE I. AVERAGE ROBOT-OBSTACLE DISTANCE ERROR AND MOVING SPEED OF DIFFERENT FCS IN THE TRAINING ENVIRONMENT.

Metrics	Speed (m/s)	Error (m)
best-location FC	0.50	0.016
best-speed FC	0.57	0.026
Single-Objective FC [2]	0.50	0.190



(a) (b)

Fig. 9. Trajectories of the robot in an unknown, test environment using (a) the best-location and (b) the best-speed FCs, where each square measures 0.5 m × 0.5 m.

VI. SIMULATIONS

This section presents simulation results of the proposed OBF, TS, and fast navigation methods, as well as comparisons with various methods.

A. *Simulation results of the OBF control*

Example 1 (fuzzy control for OBF). Fig. 1 shows the training environment created to learn the right OBF behavior. In the MO-FCACO algorithm, the population size *N* was set to 40 and $L = N/4$. The maximum number of learning iterations was set to 250. The total number of rules was set to ten, with two of them based on the expert knowledge in (4) and (5). The other eight were in the form of (3) and were learned from scratch. A total of 30 runs were performed for statistical performance evaluation. To evaluate the learning result, the two extreme FCs among the nondominated FCs in a run were selected. They were the best-location and best-speed FCs that achieved the minimum f_1 and f_2 values, respectively. Table I shows the average robot-obstacle absolute distance error and moving speed of the two FCs in the training environment.

To see the test result of these FCs in a new, unknown environment, Fig. 9 shows the trajectories of the FCs in a test environment for a total of 1200 time steps, where the FCs successfully controlled the robot in executing the OBF in the unknown environment. It was observed that the best-location FC showed smaller turning overshoots at the corners than did the best-speed FC. In contrast, the best-speed FC resulted in a longer moving distance than that of the best-position FC.

To see the advantage of the combination of expert knowledge and data-driven learning approach proposed in this paper, comparisons with the data-driven MO-FCACO algorithm and nondominated sorting GA II (NSGA-II) [46] algorithm without the expert knowledge were performed. For the two data-driven methods, the number of rules was also set to ten, with all rules in the form of (3). Table II shows the computation algorithm was applied to the same training

TABLE II. COVERAGE BETWEEN LEARNING THROUGH THE

PROPOSED COMBINATION METHOD AND THE DATA-DRIVEN METHODS USING VARIOUS OPTIMIZATION ALGORITHMS.

Data-driven algorithms	MO-FCACO [7]	NSGA II [46]
C(Proposed, B)	0.782	0.831
C(B, Proposed)	0.133	0.075

TABLE III. COVERAGE BETWEEN LEARNING THROIUGH THE PROPOSED OBJECTIVE FUNCTION AND THOSE PROPOSED IN VARIOUS STUDIES.

Objective function	Study [14]	Study [18]
C(Proposed, B)	0.595	0.569
C(B, Proposed)	0.352	0.331

TABLE IV. COVERAGE BETWEEN LEARNING THROIUGH THE PROPOSED OBJECTIVE FUNCTION AND THOSE PROPOSED IN VARIOUS STUDIES.

Objective function	Scheduled PID	Proposed
Average time step	110.1	49.6

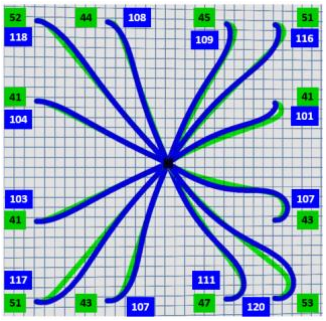
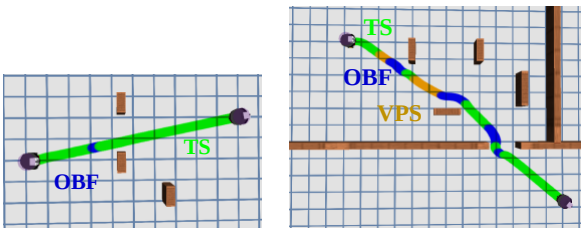


Fig. 10. TS trajectories and the total number of control time steps required to reach the target using the proposed control (green lines and boxes) and the control approach in [18] (blue lines and boxes).

coverage metric [47] between the combination method and each of the methods used for comparison (denoted by "B"). The result showed that *C*(Proposed, B) was greater than *C*(B, Proposed) for each of the algorithms used for comparison, indicating that the proposed combination method has better coverage and the advantage of incorporating the expert knowledge to improve learning performance.

To observe the advantage of including the penalty term $\delta(T_c)$ in (6) to improve the learning performance, the objective functions for evaluating the position performance in [14], [18] were also applied to the same two-objective OBF learning problem. For ease of comparison, the robot started only from point "A" in the OBF learning problem, and the ten fuzzy rules were all learned through the data-driven MO-FCACO algorithm, i.e., without using the expert knowledge. For comparison of the FCs learned through these different objective functions, all learned FCs were reevaluated using the same objective function in [14], i.e., without the term $\delta(T_c)$ in (6). Table III shows the coverage metric between the proposed approach and each of the methods used for comparison (denoted by "B"). The result showed that *C*(proposed, B) was greater than *C*(B, proposed) for any of the algorithms used for comparison, indicating that the proposed objective function has better coverage.

For comparison, the single-objective learning configuration proposed in [2] that optimized an FC using the evolutionary



(a) (b)

Fig. 11. Navigation trajectories of the robot using the proposed fast navigation scheme in environments (a) one and (b) two in Example 3, where each square measures 0.5m × 0.5 m.

TABLE V. NAVIGATION TIME AND TRAVELLED DISTANCES OF DIFFERENT SCHEMES FOR THE TWO ENVIRONMENTS IN EXAMPLE 3, WHERE THE NAVIGATION TIME OF A FAILED FC IS THE TIME THE ROBOT COLLIDES WITH AN OBSTACLE.

Design Methods		Expert [11]	Naïve [11]	GA [11]	PSO [11]	Proposed
Env. one	time (s)	17	25	27.1	25.4	8.8
	distance (cm)	528	Failed	531	542	487
Env. two	time (s)	45	25	32	21	16.9
	distance (cm)	1025	Failed	Failed	Failed	821

environment. In this comparison, ten rules without *a priori* knowledge were learned using the proposed method and [2]. Table I shows that the two FCs designed using the proposed method achieved nearly one-tenth of the distance error, and reached the speed that was higher than or equal to that in [2].

B. Simulation results of the TS control

Example 2 (TS control performance). This example shows the control performance of the fuzzy PID/PD controller in the TS. Fig. 10 shows the robot trajectories in executing the TS behavior using the proposed controllers, where the robot starts from different positions and orientations and reaches the same target. For comparison, Fig. 10 also shows the robot trajectories sing the scheduled PID control approach in [18]. Like most previous studies [16], [17], the approach in [18] used a single PID controller to control the speed of the robot. For the orientation control, a scheduled PID controller was proposed in [18] to improve control performance. Table IV shows the average number of control steps to reach the target using different methods. The average total number of control times of the proposed fuzzy PID/PD control approach was smaller than half of that in [18], which shows the superiority of the fuzzy PID/PD controller over PID controller.

C. Simulation results of the fast navigation scheme

Example 3 (fast navigation in convex maps). This example shows the performance of the fast navigation scheme in two unknown environments containing only convex obstacles, as shown in Fig. 11, studied in [11]. The location-speed FC that achieved the minimum $f_1 + f_2$ (achieving a compromise of the best-position and best-speed FCs) in Example 1 was selected for executing the OBF behavior. Fig. 11 shows the successful navigation trajectories using the proposed scheme. In this and the following figures, the TS, OBF, and VPS behaviors were represented by green, blue, and orange lines, respectively. Table IV shows the total navigation time and travelled distance.

For comparison, Table V also shows the navigation result of the approach reported in [11]. The approach consists of two fuzzy modules: a navigation behavior module and an avoiding

obstacles module. Four different methods in designing the FCs in the two modules are proposed, including the naive FCs, and their enhancements using GA, PSO, or expert knowledge [11]. Table V shows that in environment 1, the proposed navigation scheme achieved shorter navigation time and smaller travelled distance. In environment 2, among the methods used for comparison, only the FC-expert succeeded in navigating the robot. The navigation time of the proposed method was only approximately one-third of that in the FC-expert. For the GA and PSO enhanced methods in [11], the FC in the avoiding obstacles module was trained to avoid obstacles in certain TS directions instead of following obstacle boundaries. Therefore, the trained FC may fail to bypass new obstacles in a new environment, as shown in Table V.

Example 4 (fast navigation in concave maps). This example shows the performance of the fast navigation scheme in an unknown environment with concave obstacles (as shown in Fig. 12) and an unknown maze environment (as shown in Fig. 13). Fig. 12 and Fig. 13 (a) show the navigation trajectories in these environments using the proposed navigation scheme, where the robot successfully reached the target in these simulations without becoming stuck in dead cycles. In Fig. 13(a), the robot passed the same point "c" twice and performed different behaviors because the criterion $d_2 < d_1$ introduced in Section IV-C for dead-cycle avoidance was satisfied only when the robot first passed the point.

For the purpose of comparison, the online navigation schemes for local path planning proposed in [10] and [18] were also applied to the same environment in Fig. 12 by using the same FCs. Fig. 12 shows the trajectories of the two navigation schemes used for comparison. The navigation scheme in [10] faced the dead-cycle problem around U-shaped obstacles, as shown in Fig. 12. The scheme in [18] successfully navigated to the robot to the target, but the total number of navigation time steps was 256 and was longer than 182 obtained using the proposed scheme. The result showed the significant effect of the proposed VPS behavior in reducing the time needed to reach the target.

In another comparison, the online local path planning scheme using the SLAM-based DWA-A* algorithm, available as a navigation package in the ROS [43], was also applied to the unknown environment in Fig. 13(a). Fig. 13(b) shows the trajectory of the navigation scheme, where the robot moved back to a visited space around area "A" instead of passing through the passage around area "B". As a result, the travelled distance using the DWA-A* algorithm was much larger than that using the proposed navigation scheme. In the ROS simulation environment, a 360^o laser range scanner was mounted on the robot for SLAM. This scanner helped to sense the whole surrounding environment and avoid going deeper to a U-shaped environment. The proposed scheme considered the scanner covering only 180^o sensing range in simulations and experiments. Therefore, the robot may move with a larger local distance to escape a U-shaped environment. The proposed fast navigation scheme can be further modified for

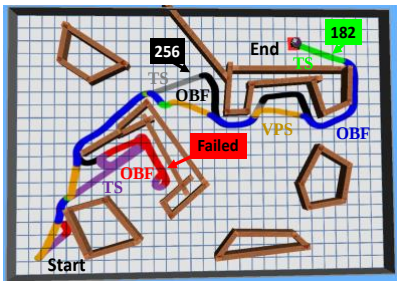
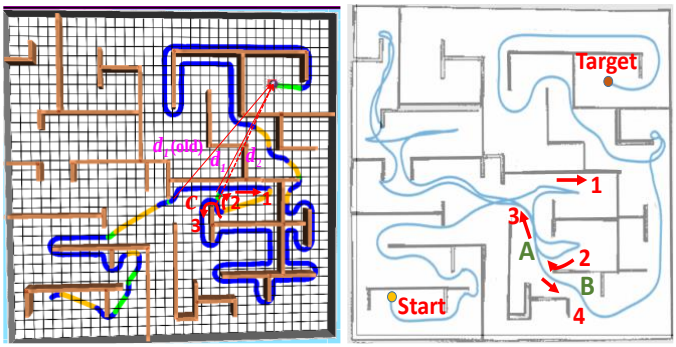


Fig. 12. Navigation trajectories of the robot using the proposed fast navigation scheme (in green, orange, and blue colors) and navigation schemes of [18] (in black and gray) and [10] (in red and purple colors) in an unknown environment in Example 4.



(a) (b)

Fig. 13. Navigation trajectories of the robot using the (a) proposed (b) DWA-SLAM navigation schemes in Example 4, where the number indicates the time order of trajectories.

performance improvement if a 360^o laser range scanner is used instead.

VII. EXPERIMENTS

This section presents experimental results of localizing and navigating the robot in real environments with static and moving obstacles to show the accuracy of the localization approach and the performance of the fast navigation scheme. Considering the differences between real and simulation robots, such as the inertia of movement and the additionally carried objects on the real robot, the moving speed of the FC in the OBF behavior in the experiments was set to be half of that in the simulations.

Example 5 (navigation with static obstacles). This example considers navigating the real robot using the proposed fast navigation scheme in an unknown environment with static obstacles. Fig. 14 shows screenshots of the navigation result in the environment. Fig. 15(a) shows the navigation trajectory of the robot, where the trajectory was recorded using the proposed localization approach. The result showed that the robot moved toward the subgoals found in the two obstacles to shorten the navigation path, where the first and second VPS behaviors were switched from the OBF and TS behaviors, respectively. Fig. 15 shows that interruptions or distortions of the StarGazer sensor measurements happened at some control time steps and that the proposed localization method corrected those failed measurements. The average root-mean-squared errors (RMSEs) of the localization and control accuracy of the final robot position were determined over ten experimental runs, and the

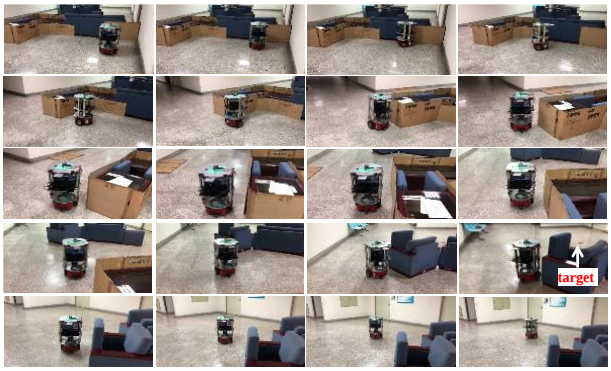
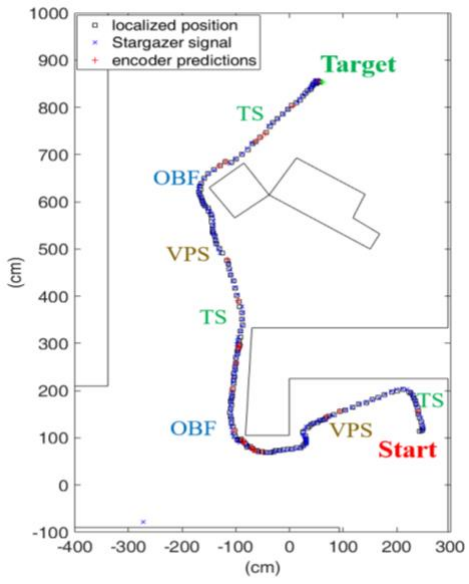
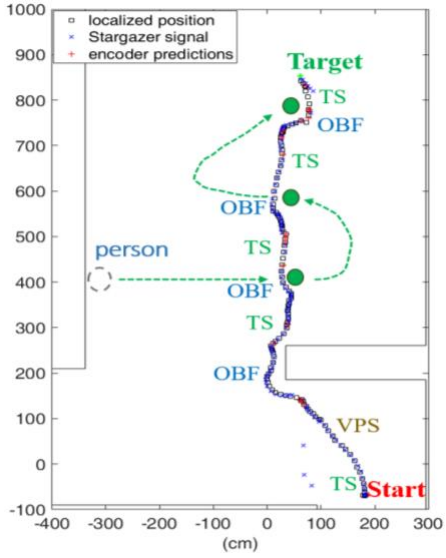


Fig. 14. Screenshots of navigating the real robot in an unknown test environment using the proposed fast navigation scheme in Example 5.



(a)



(b)

Fig. 15. Stargazer sensor measurements (×), the predicted positions using the encoder when interruptions in the sensor occur (+), and the localized positions (□) determined using the proposed particle filter-based localization method in (a) Example 5 and (b) Example 6, where the person walking trajectory is denoted as a dotted line.

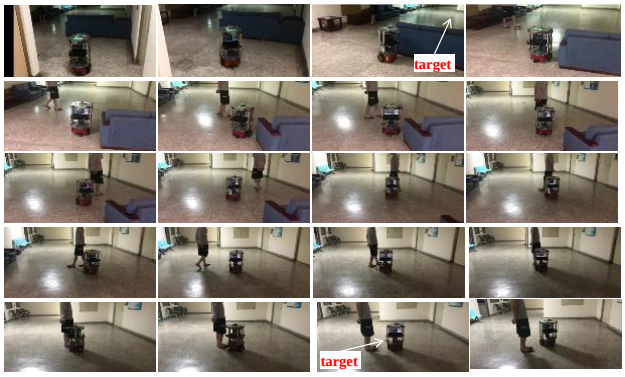


Fig. 16. Screenshots of the navigation result when the robot continuously met a moving obstacle (a person) while executing the TS behavior in Example 6.

ground truth was obtained from manual measurement. For the localization of the robot, the result showed a small average RMSE of 3.6 cm (STD=2.8 cm) between the localized and actual positions. For the control of the robot, the result also showed a small average RMSE of 4.5 cm (STD=2.1) between the actual robot position and the target.

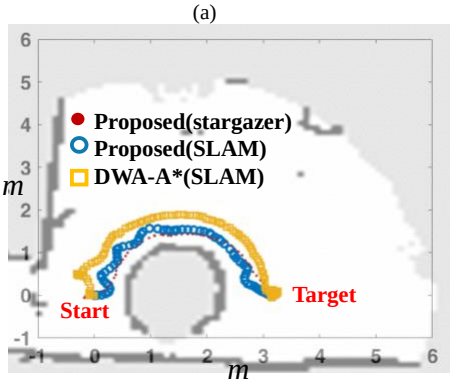
Example 6 (navigation with moving obstacles). This example presents the experimental result of navigating the robot in an unknown environment with a moving obstacle, where a person continued to walk toward the front of the robot after it detoured around the person. Fig. 16 shows screenshots of the navigation result in the environment. Fig. 15(b) shows the navigation trajectory of the robot. In this experiment, the robot initially switched from the TS behavior to the VPS behavior to shorten the route and detoured around a sofa. When the robot was executing the TS behavior, a person walked toward the moving robot and stopped in front of it. When the robot met the person at different locations, it switched the original TS behavior to the OBF behavior and successfully detoured around the person. Finally, the robot successfully reached the target.

Example 7 (navigation with SLAM). This example considers navigating the real robot in two unknown environments using the proposed navigation scheme with two different localization methods and the SLAM-based DWA-A* navigation package in the ROS [43]. Figs. 17(a) and 18(a) show the two environments, where the later contains a narrow pathway. For the proposed navigation scheme in each environment, two experiments using the proposed Stargazer-based and SLAM localization methods were performed. For the SLAM localization, a 360^{o} laser range scanner RPLIDAR A2 was mounted on the robot and the SLAM package in the ROS was used. Figs. 17 (a) and 18(a) show screenshots of the navigation results in the two environments using the proposed localization and navigation scheme. Figs. 17(b) and 18(b) show the navigation trajectories of the robot using the proposed navigation scheme with the two localization methods. The results showed that the robot successfully reached the target in all experiments. Fig. 18 also shows that the robot successfully moved through the narrow pathway to reach the target. Table VI shows the average path length, navigation time, and localization error (RMSE) using the proposed navigation scheme with the two localization methods, where the average was determined over five experimental runs. The result showed that under the same

TABLE VI. NAVIGATION PERFORMANCES USING DIFFERENT LOCALIZATION AND NAVIGATION METHODS IN EXAMPLE 7.

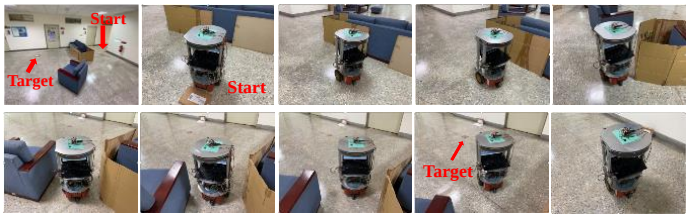
Localization Methods		SLAM		Stargazer
Navigation Scheme		DWA-A*	Proposed	Proposed
Environment 1	Time (s)	39	28	26
	Path Distance (cm)	771	537	528
	Localization error (cm)	27.5		6.5
Environment 2	Time (s)	66.8	32.6	29.6
	Path Distance	1256	714	693
	Localization error (cm)	24.9		4.6

(a)

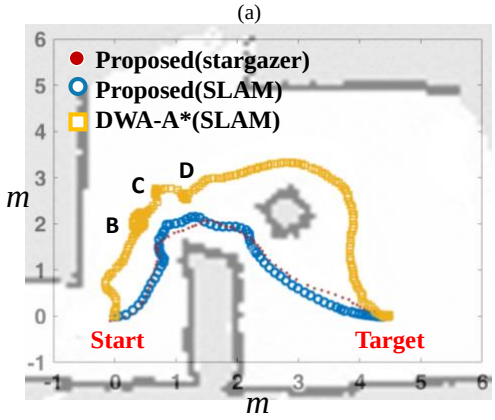


(b)

Fig. 17. (a) Screenshots of navigating the robot using the proposed localization method and navigation scheme in the first environment in Example 7. (b) Trajectories of the robot using different localization methods and navigation schemes in the map built through SLAM.



(a)



(b)

Fig. 18. (a) Screenshots of navigating the robot using the proposed localization method and navigation scheme in the second environment in Example 7. (b) Trajectories of the robot using different localization methods and navigation schemes in the map built through SLAM.

navigation scheme, the Stargazer-based localization achieved smaller localization error than the SLAM localization method.

For the purpose of comparison, the SLAM-based DWA-A* navigation scheme was also applied to navigate the robot in the two environments. In this scheme, the parameters such as the inflation radius [43] for robot-obstacle clearance setting was selected so that the robot can move along an obstacle boundary as close as possible. Fig. 17 (b) and Fig. 18(b) show the navigation trajectories in the two environments. In contrast to the proposed navigation scheme, Fig. 17(b) shows that the robot moved along the obstacle with a larger robot-obstacle distance when the DWA-A* was used. Fig. 18 (b) shows that the robot turned around at point "B" and moved backward at points "C" and "D" and finally the DWA-A* navigated the robot in bypassing the sofa to reach the target instead of moving through the narrow pathway, which caused a longer navigation path. Table VI shows the navigation performances in the two environments. The DWA-A* navigation scheme showed longer navigation time and larger travelled distance than the proposed navigation scheme when the same SLAM localization method was used in the two environments.

VIII. CONCLUSION

A new navigation scheme of a wheeled robot controlled by a new fuzzy control approach in unknown environments is proposed in this paper. The FC is used because of the explainable fuzzy rule expression ability in it. To take advantage of this ability, this paper proposes the automatic learning of an FC based on the combination of human expert rules and a data-driven multiobjective learning approach to control a real robot in executing the OBF behavior. New objective functions and constraints are proposed to improve the OBF learning performance. In comparison with the previous single-objective learning approach, the multiobjective architecture at the same time provides more diverse selections of FCs and a better control performance. For the TS, this paper proposes fuzzy PID and PD controllers that successfully control the robot to quickly move toward the target with a smooth trajectory. Based on the fuzzy controlled OBF and TS behaviors, this paper proposes a fast map-free navigation scheme, including a new VPS behavior to reduce superfluous routes, and a new behavior supervisor that successfully supervises various robot behaviors. The advantages of the proposed approaches have been verified by simulations, comparison with various methods, and experiments. In particular, high localization and control accuracies have been achieved in the experiments. The proposed navigation system can be applied to navigating robots in various environments, such as object-carrying robots in factories and home service robots.

REFERENCES

[1] A. Zhu and S. X. Yang, "Neurofuzzy-based approach to mobile robot navigation in unknown environments," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 37, no. 4, pp. 610–621, Jul. 2007.

[2] C. F. Juang and Y. C. Chang, "Evolutionary-group-based particle-swarm-optimized fuzzy controller with application to mobile-robot navigation in unknown environments," *IEEE Trans. Fuzzy Systems*, vol. 19, no. 2, pp. 379-392, April 2011.

[3] C. H. Hsu and C. F. Juang, "Evolutionary robot wall-following control using type-2 fuzzy controller with species-DE-activated continuous ACO," *IEEE Trans. Fuzzy Systems*, vol. 21, no. 1, pp. 100-112, Feb. 2013.

[4] M. Algabri, H. Ramdane, H. Mathkour, K. Al-Mutib, and M. Alsulaiman, "Optimization of fuzzy logic controller using PSO for mobile robot navigation in an unknown environment," *Applied Mechanics and Materials*, vols. 541-542, pp. 1053-1061, 2014

[5] R. Zhao, D. H. Lee, and H. K. Lee, "Mobile robot navigation using optimized fuzzy controller by genetic algorithm," *Int. J. Fuzzy Logic and Intelligent Systems*, vol. 15, no. 1, pp. 12-19, March 2015.

[6] C. Kim and D. Chwa, "Obstacle avoidance method for wheeled mobile robots using interval type-2 fuzzy neural network," *IEEE Trans. Fuzzy Systems*, vol. 23, no. 3, pp. 677-687, June 2015.

[7] C. F. Juang, T. L. Jeng and Y. C. Chang, "An interpretable fuzzy system learned through online rule generation and multiobjective ACO with a mobile robot control application," *IEEE Tran. Cyber.*, vol. 46, no. 12, pp. 2706-2718, Dec. 2016.

[8] H. Omrane, M. S. Masmoudi, and M. Masmoudi, "Fuzzy logic based control for autonomous mobile robot navigation," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 9548482, pp. 1-10, 2016.

[9] M. S. Masmoudi, N. Krichen, M. Masmoudi, nd N. Derbel, "Fuzzy logic controllers design for omnidirectional mobile robot navigation," *Applied Soft Computing*, vol. 49, pp. 901–919, 2016.

[10] K. Al-Mutib and F. Abdessemed, "Indoor mobile robot navigation in unknown environment using fuzzy logic based behaviors," *Advances in Science, Technology and Engineering Systems Journal*, vol. 2, no. 3, pp. 327-337, 2017.

[11] M. Faisal, M. Algabri, B. M. Abdelkader, H. Dhahri, M. M. A. Rahhal, "Human expertise in mobile robot navigation," *IEEE Access*, vol. 6, pp. 1694-1705, 2018.

[12] L. Kong, W. He, C. Yang, Z. Li, and C. Sun, "Adaptive fuzzy control for coordinated multiple robots with constraint using impedance learning," *IEEE Trans. Cyber*., vol. 49, no. 8, pp. 2052-3063, Aug. 2019.

[13] R. H. Abiyev, N. Akkaya, I. Gunsel, "Control of omnidirectional robot using Z-number-based fuzzy system," *IEEE Trans. Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, Jan. 2019.

[14] C. F. Juang, C. H. Lin, and T. B. Bui, "Multiobjective rule-based cooperative continuous ant colony optimized fuzzy systems with a robot control application," *IEEE Trans. Cyber.*, vol. 50, no. 2, pp. 650-663, Feb. 2020.

[15] C. F. Juang and T. B. Bui, "Reinforcement neural fuzzy surrogate-assisted multiobjective evolutionary fuzzy systems with robot learning control application," *IEEE Trans. Fuzzy Systems*, vol. 28, no. 3, pp. 434-446, March 2020.

[16] H. Xiaoqian, H. Karki, A. Shukla, and Z. Xiaoxiong, "Variant PID controller design for autonomous visual tracking of oil and gas pipelines via an unmanned aerial vehicle," *Proc. 17th Int. Conf. Control, Automation and Systems*, Jeju, Korea, 2017, pp. 368-372.

[17] A. Aouf, L. Boussaid and A. Sakly, "A PSO algorithm applied to a PID controller for motion mobile robot in a complex dynamic environment," *Proc. Int. Conf. Engineering & MIS*, Monastir, 2017, pp. 1-7.

[18] C. Y. Chou and C. F. Juang, "Navigation of an autonomous wheeled robot in unknown environments based on evolutionary fuzzy control," *Inventions*, vol. 3, no. 1, pp. 1-14, Jan. 2018.

[19] V. Sood, "Autonomous robot motion control using fuzzy PID controller," *Proc. Int. Conf. High Performance Architecture and Grid Computing*, pp. 385-390, 2011.

[20] Q. Xu, J. Kan, S. Chen, and S. Yan, "Fuzzy PID based trajectory tracking control of mobile robot and its simulation in Simulink," *Int. J. Control and Automation*, vol. 9, no. 11, pp. 203-214, 2016.

[21] M. S. Jie and W. H. Choi, "Type-2 fuzzy PID controller design for mobile robot," *Int. J. Control and Automation*, vol. 9, no. 11, pp. 203-214, 2016.

[22] H. Wang, Y. Yu, and Q. Yuan, "Application of Dijkstra algorithm in robot path-planning," in *Proc. 2nd Int. Conf. Mechanic Automation and Control Eng.*, pp. 1067-1069, China, July 2011.

[23] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Science and Cyber.*, vol. 4, no. 2, pp. 100–107, July 1968.

[24] C. Zhong, S. Liu, Q. Lu, B. Zhang, and S. X. Yang, "An efficient fine-to-coarse wayfinding strategy for robot navigation in regionalized environments," *IEEE Trans. Cyber.*, vol. 46, no. 12, pp. 3157-3170, Dec. 2016.

[25] M. Kusuma, Riyanto, and C. Machbub, "Humanoid robot path planning and rerouting using A-star search algorithm," in *Proc. IEEE Int. Con. Signals and Systems*, pp. 110-115, Indonesia, July 2019.

[26] K. Daniel, A. Nash, S. Koenig, and A. Felner, "Theta*: Any-angle path planning on grids," *J. Artificial Intelligence Research*, vol. 39, pp. 533-579, 2010.

[27] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: a review survey," *IEEE Access*, vol. 2, pp. 56-77, 2014.

[28] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[29] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robotics Research*, vol. 20, no. 5, pp. 378–400, May 2001.

[30] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res*., vol. 30, pp. 846–894, 2011.

[31] D. Connell and D. Connell, "Extended rapidly exploring random tree–based dynamic path planning and replanning for mobile robots," *Int. J. Advanced Robotic Systems*, pp. 1-15, May-June 2018.

[32] A. Bakdi, A. Hentout, H. Boutami, A. Maoudj, O. Hachour, and B. Bouzouia, "Optimal path planning and execution for mobile robots using genetic algorithm and adaptive fuzzy-logic control," *Robotics and Autonomous Systems*, vol. 89, no. 1, pp. 95-109, March 2017.

[33] B. Y, Q. Tang, J. Yao, and W. Gao, "Collision-free path planning and delivery sequence optimization in noncoplanar radiation therapy," *IEEE Trans. Cyber*., vol. 49, no. 1, pp. 42-55, Jan. 2019.

[34] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," *Proc. IEEE Int. Conf. Robotics and Automation*, Sacramento, CA, USA, vol. 2, 1991, pp. 1398-1404.

[35] A. Azzabi and K. Nouri, "Path planning for autonomous mobile robot using the potential field method," in *Proc. Int. Conf. Advanced Systems and Electric Technologies*, Tunisia, Jan. 2017, pp. 389-394.

[36] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Autom. Mag*., vol. 4, pp. 23-33, March 1997.

[37] T. Liu, R. Yan, G. Wei, and L. Sun, "Local path planning algorithm for blind-guiding robot based on improved DWA algorithm," in *Proc. Chinese Control And Decision Conference*, pp. 6169-6173, China, June 2019.

[38] S. Zuo, O. Yongsheng, and X. Zhu, "A path planning framework for indoor low-cost mobile robots," in *Proc. IEEE Int. Con. Information and Automation*, pp. 18-23, China, July, 2017.

[39] S. Kuswadi, J. W. Santoso, M. N. Tamara, and M. Nuh, "Application SLAM and path planning using A-star algorithm for mobile robot in indoor disaster area, " in *Proc. Int. Electronics Symp. Eng. Tech. and Applications*, pp. 270-274, Indonesia, Oct. 2018.

[40] M. Muhtadin, R. M. Zanuar, I. K. E. Purnama, and M. H. Purnomo, "Autonomous navigation and obstacle avoidance for service robot," in *Proc. Int. Conf. Computer Engineering, Network, and Intelligent Multimedia*, pp. 1-8, Indonesia, Nov. 2019.

[41] A. Koubaa, Robot Operating System (ROS): The Complete Reference (Volume 1), Cham: Springer International Publishing, 2016.

[42] T. Zeng and B. Si, "Mobile robot exploration based on rapidly-exploring random trees and dynamic window approach," in *Proc. Int. Conf. on Control, Automation and Robotics*, pp. 51-57, China, April 2019.

[43] K. Zheng, "ROS Navigation Tuning Guide," arXiv:1706.09068, pp. 1-23, April 2019

[44] J. Kim and W. Chung, "Localization of a mobile robot using a laser range finder in a glass-walled environment," *IEEE Trans. Industrial Electronics*, vol. 63, no. 6, pp. 3616-3627, June 2016.

[45] S. Yoon, S. Park, and J. Kim, "Kalman filter sensor fusion for Mecanum wheeled automated guided vehicle localization," *Journal of Sensors*, vol. 2015, Article ID 347379, 7 pages, Jan. 2015.

[46] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, Apr 2002.

[47] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Trans. Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, Nov. 1999.