Regular Article

A Robust Mobile Robot Navigation System using Neuro-Fuzzy Kalman Filtering and Optimal Fusion of Behavior-based Fuzzy Controllers

Thi Thanh Van Nguyen, Ha Vu Le, Quang Vinh Tran

Faculty of Electronics and Telecommunications, VNU University of Engineering and Technology

Correspondence: Thi Thanh Van Nguyen, vanntt@vnu.edu.vn

Communication: received 5 August 2016, revised 29 May 2017, accepted 13 July 2017
Online publication: 30 October 2017, Digital Object Identifier: 10.21553/rev-jec.128
The associate editor coordinating the review of this article and recommending it for publication was Dr. Tran Trung Duy.

Abstract– **This study proposes a control system model for mobile robots navigating in unknown environments. The proposed model includes a neuro-fuzzy Extended Kalman Filter for localization task and a behavior-based fuzzy multi-controller navigation module. The neuro-fuzzy EKF, used for estimating the robot's position from sensor readings, is an enhanced EKF whose noise covariance matrix is progressively adjusted by a fuzzy neural network. The navigation module features a series of independently-executed fuzzy controllers, each deals with a specific navigation sub-task, or behavior, and a multi-objective optimizer to coordinate all behaviors. The membership functions of all fuzzy controllers play the roles of objective functions for the optimizer, which produces an overall Pareto-optimal control signal to drive the robot. A number of simulations and real-world experiments were conducted to evaluate the performance of this model.**

Keywords– **Mobile robots, navigation, localization, behavior-based architecture, fuzzy logic, multi-objective optimization.**

1 Introduction

Mobile robot navigation is one of the most challenging problems in robotics. Target-reaching is a fundamental navigation task for any autonomous mobile robot, for which the robot must be capable of *perceiving* its surrounding environment via meaningful data extracted from sensors, *localizing* robot's position in the environment, *planning* the path to the intended target, and *controlling* the actuators to drive the robot to reach that target [1]. Given the robot's locomotion mechanism and sensors, one must design an efficient navigation architecture and methods for localization and control.

Localization is to estimate the robot's position and orientation relative to the environment's reference frame from sensor data. Localization methods are generally divided into three categories: *dead reckoning*, *absolute positioning*, and *sensor fusion* [2]. Dead reckoning methods estimate the position by tracking the robot's trajectory since the last known position based on instantaneous speed and heading direction. However, in dead reckoning the estimation errors will be accumulated over time if no compensation is provided. Unlike dead reckoning, absolute positioning methods try to calculate the robot's relative position in the reference frame, independent from previous estimates. Though absolute positioning does not suffer from accumulative errors, this approach often requires complex computations and is dependent on structural features of the environment. The sensor fusion approach employs both relative and absolute measurements in order to achieve more accurate estimates. Several fusion techniques have been proposed, such as Kalman filtering [3], Bayesian filtering [4], and particle filtering [5]. Among them, the Extended Kalman Filter (EKF) for non-linear estimation is a powerful method which combines multi-sensor data to give optimal estimates in a statistical sense with simpler structure and less time consuming than other filers. In EKF, however, the choice of noise covariance matrices greatly affects the estimation accuracy. Due to random nature of noise, theses matrices vary from time to time and therefore are difficult to determine. In practice, noise covariance matrices are often assumed to be fixed and chosen via off-line processes. This simplification may cause the EKF to diverge in some cases.

The robot's navigation module features the integration of all components responsible for navigation tasks, including perceptual interpretation of sensor data, localization, and path planning. There are two main categories of navigation architectures: the *hierarchical* architecture and the *reactive*, or *behavior-based*, architecture [6]. The hierarchical architecture is characterized by sequential steps of sensing, planning, and acting, based on a known model of the environment. This architecture is appropriate only for static and structured environments. For robots operating in unknown environments, the behavior-based architecture is more commonly used. In behavior-based systems, a complex navigation task is split into sub-tasks, or behaviors. Each behavior, dealing with a specific problem navigation, is implemented by an independent control module. The output signals of all behavioral control modules are then combined in accordance with some global navigating objective to generate an overall control signal. The main challenge with behavior-based systems is how to combine behaviors, termed *command fusion* or *behavior coordination*, to efficiently achieve the navigation objective. Many command fusion techniques have been proposed, such as switching [7], motor schema [8], and decentralized information filter (DIF) [9], but the most popular approach employed by mobile robot navigation systems has been fuzzy-based fusion [10–14]. In a fuzzy-based system, each behavior is implemented by a fuzzy controller. The output fuzzy sets from all fuzzy controllers are combined and then defuzzified to generate the overall control signal. This approach is simple to implement and quite efficient in navigation. The fusion, however, is not optimal as defuzzification processes often result in different types of control values [15, 16]. To overcome this weakness, a method based on multi-objective optimization theories, called MOASM, was proposed [17]. This method uses a set of objective functions, each is assigned to one behavior, to transform control signals into values reflecting the grades of behavioral objectives. A multi-objective optimization process is applied to find the solution which best maximizes all the outputs of objective functions. The main advantage of this method is its theoretical approach to secure the optimality of the found solutions. However, the lack of a framework for designing objective functions, which are usually complicated, limited its practical use.

In this study, a robust navigation system for mobile robots operating in unknown environments is proposed. This closed-loop navigation system places a localization module on the feedback path that estimates the robot's position from sensor readings. The localization module is basically a neuro-fuzzy Kalman filter, called FNN-EKF, an enhanced Extended Kalman Filter (EKF) in which the noise covariance matrix is adjusted by a Fuzzy Neural Network (FNN) at each iteration. The FNN itself is an improved fuzzy controller whose membership functions are tuned by a neural network. The proposed neuro-fuzzy Kalman filter can adapt better than the original EKF to varying noise conditions.

A behavior-based fuzzy multi-controller navigation method, called BBFM, was also proposed. Each behavior of this behavior-based navigation system is represented by a fuzzy controller which implements only procedures for fuzzification and fuzzy inference. Each output of a fuzzy controller is considered as the value of an objective function representing the grade of a behavior's objective. The outputs of all fuzzy controllers are then used as inputs for a multi-objective optimizer which is tasked with finding the optimal value for the overall control signal.

This paper has four more sections. Section II presents the structure and components of the proposed navigation system. Section III provides simulation results and analyses of system performance in several operational scenarios. Section IV shows some experiments with the system implemented in a real robot. Section V concludes this work.

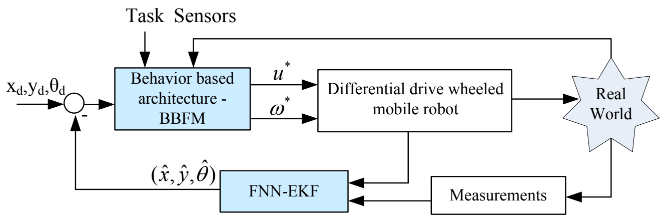

Task Sensors
x_d, y_d, θ_d
-
Behavior based architecture - BBFM
u^*
ω^*
Differential drive wheeled mobile robot
Real World
$(\hat{x}, \hat{y}, \hat{\theta})$
FNN-EKF
Measurements

Figure 1: The mobile robot's feedback navigation system.

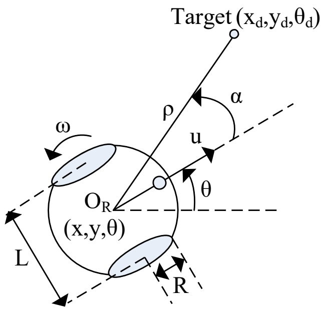


Figure 2: The differential-drive wheeled mobile robot and its parameters.

2 Proposed navigation system

In this section, the overall structure of the proposed navigation system is described. After that, detailed implementation of each primary module is presented.

2.1 The overall structure

Given the robot together with its sensory system, the mission of the navigation system is to navigate the robot from an initial position to a desired target without colliding with obstacles and getting trapped in any area of an unknown environment. To carry out this task, a control system with two primary modules, the positioning and the behavior-based control, is proposed as shown in Figure 1. Details of each module are presented following the description of the model of mobile robot.

The model of mobile robot we employ for evaluating the proposed navigation system is a differential-drive wheeled robot with non-holonomic constraints. Its parameters are shown in Figure 2, where R is the wheel diameter, L is the distance between two wheels, and the (x, y, θ) represents the position and heading orientation of the robot. For the sake of convenience, whenever we refer to the robot's position in this paper, we mean its position and heading orientation. The kinematic equation in discrete-time domain of the robot is presented by [18]:

$$\begin{cases} x_k = x_{k-1} + u_{k-1} T_s \cos \theta_{k-1} \\ y_k = y_{k-1} + u_{k-1} T_s \sin \theta_{k-1} \\ \theta_k = \theta_{k-1} + \omega_{k-1} T_s. \end{cases} \tag{1}$$

in which, T_s is the sampling period, u_k and ω_k are respectively the tangential velocity and angular velocity at the sampling time of k.

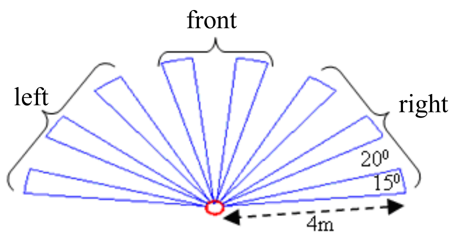


Figure 3: Arrangement of ultrasonic sensors on the robot.

To sense the environment, the robot is equipped with eight ultrasonic sensors with the measuring range from 0.04 m to 4 m. They are clustered into three groups of left, right, and front as shown in Figure 3. The measuring value of each group is the minimum value of all sensors in that group:

$$\begin{aligned} d_r &= \min(d_1, d_2, d_3), \\ d_f &= \min(d_4, d_5), \\ d_l &= \min(d_6, d_7, d_8), \end{aligned} \tag{2}$$

where d_i is the distance to obstacles measured by sensor i.

2.2 Positioning module

In order to reach the target, the robot first needs to know its position and orientation in the environment. Those information can be obtained by using measurements of the optical encoders and compass sensor. However, the measurements may subject to noise resulting in inaccurate positioning. To deal with this problem, the extended Kalman filter (EKF) has been proven to be effective. It incorporates information of the robot kinematics and sensor measurements so that the estimated result is more reliable. The incorporation is conducted via two steps: the *time update* to predict the robot position and orientation based on the kinematic model, and the *data update* to correct the prediction based on sensory measurements.

Let $\mathbf{x} = (x, y, \theta)^T$ be the state vector. This state can be observed by some measurements, $\mathbf{z}$. These measurements are described by a function, h, of the robot state and a Gaussian noise process, $\mathbf{v}$. Denoting the function (1) as f, with an input vector $\mathbf{u} = (u, \omega)^T$ and the process noise $\mathbf{w}$. The robot system then can be described by:

$$\begin{aligned} \mathbf{x}_k &= f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}) \\ \mathbf{z}_k &= h(\mathbf{x}_k, \mathbf{v}_k) \end{aligned} \tag{3}$$

where $k \in \mathbb{N}$, $\mathbf{x}$, $\mathbf{w} \in \mathbb{R}^n$, $\mathbf{z}$, $\mathbf{v} \in \mathbb{R}^m$, $\mathbf{u} \in \mathbb{R}^l$, $(\mathbf{v}, \mathbf{w})$ are Gaussian, uncorrelated, white, with zero-mean and covariance matrix (P, Q) respectively.

- *Prediction with time update equations*

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, 0), \tag{4}$$

$$P_k^- = A_{k-1} P_{k-1} A_{k-1}^T + Q_{k-1}, \tag{5}$$

where $\hat{\mathbf{x}}_k^-$ is the *priori* state estimate at step k given knowledge of the process prior to step $k-1$, P_k^- denotes the covariance matrix of the *priori* prediction error.

- *Correction with measurement update equations*

$$K_k = P_k^- H_k^T [H_k P_k^- H_k^T + R_k]^{-1}, \tag{6}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K_k[\mathbf{z}_k - h(\hat{\mathbf{x}}_k^-, 0)], \tag{7}$$

$$P_k = [I - K_k H_k] P_k^-, \tag{8}$$

where $\hat{\mathbf{x}}_k \in \mathbb{R}^n$ is the *posteriori* state estimate at step k given measurement $\mathbf{z}_k$, K_k is the Kalman gain, and P_k is the covariance matrix of the *posteriori* estimate error.

In implementation, the efficiency of the EKF depends, to a great extend, on the accuracy of noise covariance matrices Q and R to be determined. In our system, we model the process noise as being proportional to the angular velocities of the left and right wheels. This approach is adequate provided that the systematic errors are eliminated by the calibration process. The variances of the process noise then equal to $\delta\omega_L^2$ and $\delta\omega_R^2$, where δ is a constant determined by experiments. Thus the covariance matrix Q_k is given by:

$$Q_k = \begin{bmatrix} \delta\omega_L^2(k) & 0 \\ 0 & \delta\omega_R^2(k) \end{bmatrix}. \tag{9}$$

With covariance matrix R, we determine it based on the residual between the actual and the predicted measurements, $\mathbf{r}_k = \mathbf{z}_k - h(\hat{\mathbf{x}}_k^-, 0)$. This residual, gained by K, is the correction factor to form the posterior estimate $\hat{\mathbf{x}}_k$ from the prior estimate $\hat{\mathbf{x}}_k^-$. It also reflects the accuracy of the estimation value. A small value of $\mathbf{r}_k$ implies a good estimation because the predicted measurement is closed to the real one. Consequently, we can exploit $\mathbf{r}_k$ as an indicator to adjust R_k. Specifically, we present R_k as:

$$R_k = R_{k-1} + \Delta R_k, \tag{10}$$

where ΔR_k is an adjustable factor. Let S_k be the covariance of $\mathbf{r}_k$:

$$S_k = H_k P_k^- H_k^T + R_k, \tag{11}$$

and C_k be the average covariance of $\mathbf{r}$ in N recent estimations:

$$C_k = \frac{1}{N} \sum_{j=k-N+1}^{k} \mathbf{r}_j \mathbf{r}_j^T. \tag{12}$$

The value of ΔR_k is then adjusted in accordance to the difference, D_k, between S_k and C_k:

$$D_k = S_k - C_k. \tag{13}$$

If $D_k \cong 0$, ΔR_k is kept unchanged. If $D_k > 0$, ΔR_k is decreased. Otherwise, ΔR_k is increased. The amount of decrease or increase should depend on the value of D_k. We thus design a fuzzy controller for the adjustment.

The fuzzy controller has three input variables $D_k = \{\text{Positive(P), Zero(Z), Negative(N)}\}$, the output variable $\Delta R_k = \{\text{Decrease(D), Maintain(M), Increase(I)}\}$. The membership functions of input/output variables are defined by Gaussian (G) and Sigmoid (S) fuzzy set

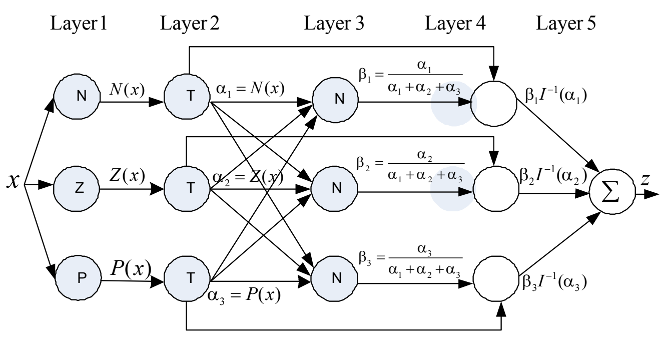


Figure 4: The fuzzy neural network.

such as:

$$\begin{aligned} \text{Gauss}(x) &= \frac{e^{-(x-c_j)^2}}{2\sigma_j^2}, \\ \text{Sigmoid}(x) &= \frac{1}{1+e^{-a_i(x-b_i)}}, \end{aligned} \qquad (14)$$

where a_i, b_i, c_j, σ_j are the distinct values for each membership function and learned by fuzzy neural network. The defuzzification is accomplished by the centroid method:

$$\phi = \frac{\sum x_i \mu(x_i)}{\sum \mu(x_i)} \qquad (15)$$

where x_i is the ith domain value and $\mu(x_i)$ is the truth membership value for that domain point.

Though the fuzzy controller provides an adaptive adjustment of ΔR_k, the determination of coefficients (a_i, b_i) and (c_j, σ_j) during implementation still requires manual customization. We overcome this limitation by designing a neural network that automatically tunes (a_i, b_i) and (c_j, σ_j). The neural network consists of five layers as shown in Figure 4. Given crisp training set $\{(D_1, \Delta R_1), (D_2, \Delta R_2), ..., (D_M, \Delta R_M)\}$, the measure of error for the mth training pattern is defined as following equation

$$E_m = \frac{1}{2}(z_m - \Delta R_m)^2, m = 1...M \qquad (16)$$

where z_m is the computed output form of the fuzzy neural network. The parameters a_i, b_i, c_j, σ_j, $i = 1, 2, 3, j = 1, 2$ are learned by the steepest descent method as follows

$$\begin{aligned} a_i(t+1) = a_i(t) - \eta \frac{\partial E_m}{a_i}, b_i(t+1) = b_i(t) - \eta \frac{\partial E_m}{b_i}, \\ c_j(t+1) = c_j(t) - \eta \frac{\partial E_m}{c_j}, \sigma_j(t+1) = \sigma_j(t) - \eta \frac{\partial E_m}{\sigma_j}. \end{aligned} \qquad (17)$$

where $\eta > 0$ is the learning constant and t is the number of the adjustments. These learned coefficients are then used to update membership functions of the fuzzy controller. We call our approach the neuro-fuzzy extended Kalman filter.

2.3 Control module

The position of robot determined by the positioning module is fed to the control module together with the target and environment information to generate control signals as shown in Figure 1. The control module is implemented using the behaviour-based approach that subdivides the navigation task into small and easy-to-manage behaviours. For our task assigned, the behaviours include: (i) reaching the target from an arbitrary position, (ii) avoiding obstacles. Each behaviour is implemented by a fuzzy controller so that it can cope with uncertainties and incompleteness of sensory information. The coordination between behaviours is accomplished by a multi-objective optimization process. The rationale is to achieve an optimal solution that can guarantee a suitable trade-off between a multitude of conflicting behaviours' objectives.

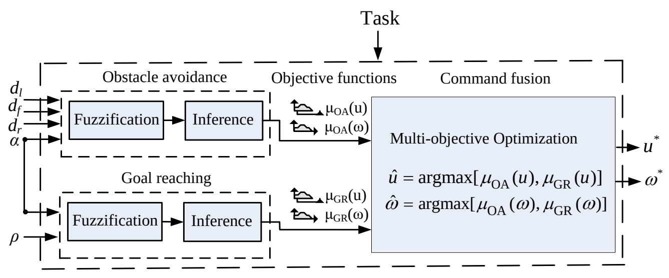


Figure 5: The BBFM navigation module.

In multi-objective optimization, the objective of each behaviour is quantified through a function, called the objective function, that assigns to each input a value reflecting its objective' desirability. Interestingly, if we remove the defuzzification procedure in the design process of fuzzy controllers, the output of each behavior will be a fuzzy membership function that encrypts the semantic meaning of the objective function. This way, the fuzzy control and multi-objective optimization can be combined as shown in Figure 5.

In this section, we first describe the implementation of behaviours using fuzzy logic. We then present our approach to fuse them using multi-objective optimization.

2.3.1 Obstacle avoidance: Due to non-holonomic constraints, we first define three additional variables for the implementation of behaviours: ρ defined by (18) is the distance from the center of the robot to the target; α defined by (19) is the angle between the robot heading and the vector connecting the robot center with the target.

$$\rho = \sqrt{(x_d - x)^2 + (y_d - y)^2)} \qquad (18)$$

$$\alpha = \arctan(y_d - y, x_d - x) - \theta, \alpha \in [-\pi, \pi]. \qquad (19)$$

The obstacle avoidance behaviour then consists of four input variables d_r, d_f, d_l, and α, and two output variables u and ω as shown in Figure 5. Their linguistic terms and membership functions are defined as shown in Figure 6.

Twenty-eight control rules defined for the behavior are presented in Table **??**. Let $\mu_{R_{OA,k}}(u)$ and $\mu_{R_{OA,k}}(\omega)$ be respectively the inference results for u and ω of the kth rule in this table. The implication results according

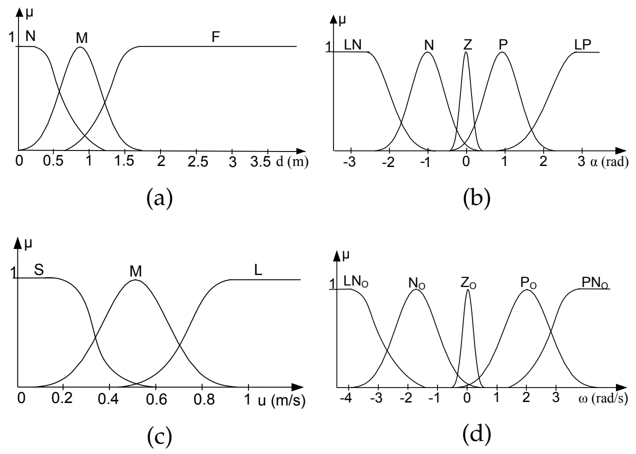


Figure 6: The linguistic terms and membership functions of input and output variables of the obstacle avoidance behavior: (a) d_l, d_f, d_r; (b) α; (c) u; (d) ω.

Table I: Rules for Obstacle Avoidance

Collisions	Rule	Input				Output	
		d_l	d_f	d_r	α	u	ω
	1	N	N	F		S	Po
	2	F	N	N		M	Po
	3	M	N	N		M	Po
	4	F	N	M		N	LPo
	5	N	N	F		M	LNo
	6	N	N	M		M	No
	7	F	N	F		M	LPo
	8	M	N	F		M	No
	9	F	N	M		M	Po
	10	N	M	N		M	Po
	11	N	N	N		S	Po
	12	M	N	M		S	Po
	13	N	M	M		M	No
	14	N	M	F		M	No
	15	N	F	M		M	No
	16	N	F	F	LN	S	LNo
	17	N	F	F	N	S	No
	18	N	F	F	Z	L	Zo
	19	N	G	G	LP	L	Zo
	20	N	F	F	P	L	Ko
	21	M	M	N		M	Po
	22	F	M	N		M	Po
	23	M	F	N		M	Po
	24	F	F	N	LN	L	Zo
	25	F	F	N	N	L	Zo
	26	F	F	N	Z	L	Zo
	27	F	F	N	LP	S	LPo
	28	F	F	N	P	S	LPo

S:Small, M:Medium, F:Far, Z/Zo:Zero,LN/LNo:Large Negative, N:Near, P/Po:Positive, N/No:Negative, LP/LPo:Large Positive

to the max-min method are then given by:

$$\begin{aligned}\mu_{R_{OA}}(u) &= \max(\mu_{R_{OA,1}}(u), \mu_{R_{OA,2}}(u), \ldots, \mu_{R_{OA,28}}(u))\\ \mu_{R_{OA}}(\omega) &= \max(\mu_{R_{OA,1}}(\omega), \mu_{R_{OA,2}}(\omega), \ldots, \mu_{R_{OA,28}}(\omega)).\end{aligned} \tag{20}$$

2.3.2 Goal reaching: The objective of goal-reaching behavior is to control the robot to reach the target in the shortest time. The behavior uses two input and

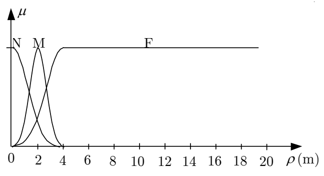


Figure 7: The linguistic terms and membership function of ρ.

Table II: Rules for Goal Reaching

Rule	Input		Output	
	ρ	α	u	ω
1	N	Z	S	Zo
2	N	N	S	No
3	N	LN	S	LNo
4	N	P	S	Po
5	N	LP	S	LPo
6	M	Z	M	Zo
7	M	N	M	No
8	M	LN	M	LNo
9	M	P	M	Po
10	M	LP	M	LPo
11	F	Z	L	Zo
12	F	N	L	No
13	F	LN	L	LNo
14	F	P	L	Po
15	F	LP	L	LPo

S:Small, N:Near, M:Medium, F:Far, Z/Zo:Zero, N/No:Negative,LN/LNo: Large Negative, P/Po:Positive, LP/LPo:Large Positive, L:Large

two output variables as shown in Figure 5. Three of them including the deflection angle α and the velocities u and ω have the same definition of linguistic terms and membership functions as in the obstacle avoidance behavior. The fourth variable, ρ, has the linguistic terms and membership functions defined as shown in Figure 7.

The behavior has 15 rules defined as in Table II. Let $\mu_{R_{GR,k}}(u)$ and $\mu_{R_{GR,k}}(\omega)$ be respectively the results of the kth rule in the table for output variables u and ω by using Equation (**??**). The implication results for u and ω according to the max-min method are then given by:

$$\begin{aligned}\mu_{R_{GR}}(u) &= \max(\mu_{R_{GR,1}}(u), \mu_{R_{GR,2}}(u), \ldots, \mu_{R_{GR,15}}(u))\\ \mu_{R_{GR}}(\omega) &= \max(\mu_{R_{GR,1}}(\omega), \mu_{R_{GR,2}}(\omega), \ldots, \mu_{R_{GR,15}}(\omega)).\end{aligned} \tag{21}$$

2.3.3 Behavior coordination: The behavior coordination is carried out by using multi-objective optimization. Let $\mu_i(y)$ be the ith objective function, y be an output control signal ($y = u$ or $y = \omega$), Y be the domain of y, and N be the number of objective functions. The optimal value of each output control signal is then the solution of the following equation:

$$\widehat{y} = \text{argmax}[\mu_1(y), \mu_2(y), \ldots, \mu_N(y)]. \tag{22}$$

According to the theory of multi-objective optimization, there might not exist the optimal solution, $\widehat{y}$, of Equation (22), but only the "*good enough*" solution, y^*, which is the best fit for all objectives. This solution is called the Pareto-optimal solution or non-dominated solution defined as follows: y^* is the Pareto-optimal solution of Equation (22) if there does not exist any $y \in Y$ such that $\mu_i(y) > \mu_i(y^*)$ for at least one i and $\mu_j(y) \geq \mu_j(y^*)$ for all j. In other words, the Pareto-optimal solution is the one in which there is not other solution that improves an objective without resulting in the deterioration of at least another objective. In our system, the optimal values of the overall control signal are determined by the output membership functions of (20) and (21) as follows:

$$\begin{aligned} \widehat{u} &= \operatorname{argmax}[\mu_{R_{OA}}(u), \mu_{R_{GR}}(u)], \\ \widehat{\omega} &= \operatorname{argmax}[\mu_{R_{OA}}(\omega), \mu_{R_{GR}}(\omega)]. \end{aligned} \tag{23}$$

The lexicographic method used to find the Pareto-optimal solutions of (23) are carried out as follows:

- Sorting all behaviors in descending order of importance: obstacle avoidance and goal reaching.
- Sequentially solving equations P_i by using discrete values of u and ω on their domains U and W until a unique solution is obtained, or all equations are solved:

$$\begin{aligned} u^* &: \begin{cases} P_1 : \max\limits_{u \in U}[\mu_{R_{OA}}(u)], \\ P_2 : \max\limits_{u \in U_1}[\mu_{R_{GR}}(u)], \\ U_1 = \{u | u \text{ solves } P_1\} \end{cases} \\ \omega^* &: \begin{cases} P_1 : \max\limits_{\omega \in W}[\mu_{R_{OA}}(\omega)], \\ P_2 : \max\limits_{\omega \in W_1}[\mu_{R_{GR}}(\omega)], \\ W_1 = \{\omega | \omega \text{ solves } P_1\} \end{cases} \end{aligned} \tag{24}$$

- If more than one Pareto-optimal solution are obtained, the one with the greatest value of u and the smallest value of ω is chosen.

3 Simulations

Simulations have been conducted to evaluate the efficiency of the proposed navigation system. Firstly, the improved FNN-EKF is compared to the conventional EKF for positioning module. Secondly, the BBFM module is independently evaluated by comparing to two other popular navigation modules including the MOASM [17] and CDB [11] without using FNN-EKF. Finally, due to non systematic error, the navigation system with BBFM &FNN-EKF as in Figure 1 is used to perform the navigation task.

All cases are simulated in Matlab and use the same robot configuration. Its mechanical parameters are set as follows: $R = 0.085$ m, $L = 0.265$ m, $u \in [0, 1.3]$ m/s, and $\omega \in [-4.3, 4.3]$ rad/s. The ultrasonic sensors have the sensing range from 0 m to 4 m and the radiation cone of 15^0. They are arranged in front of the robot as shown in Figure3 to cover the range of 160^0. The universe of discourse of ρ is in the range of [0, 20]. Details and results of each case are presented as follows.

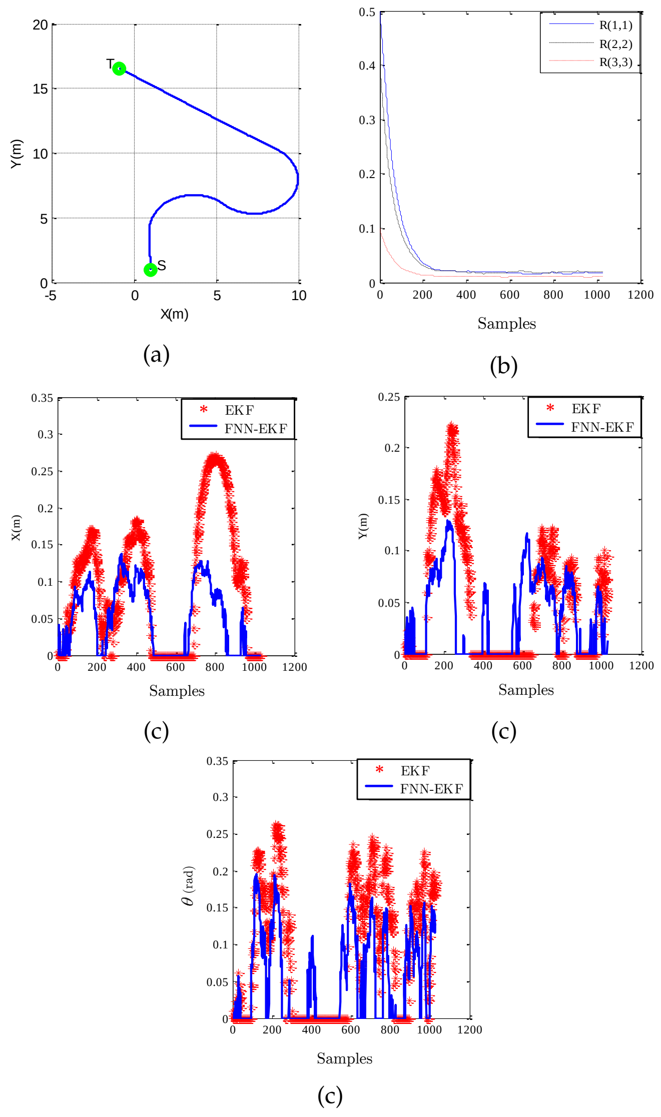


Figure 8: Positioning by FNN-EKF: (a) True path, (b) R , (c) Deviation in x direction, (d) Deviation in x direction, (e) Deviation in x direction

3.1 Comparing the FNN-EKF to the EKF

The input noise covariance matrix Q is defined as (9) where $\delta = 0.01$ is determined by experiments. The reference matrix R_{ref} is also determined by experiments as (25). The unknown assumed matrix is initially chosen as R. The FNN-EKF tunes R by FNN while the conventional EKF does not.

$$R_{ref} = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.018 \end{bmatrix}; R = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.4 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} \tag{25}$$

A true path for positioning is presented Figure 8(a). After some interations, the matrix R tuned by FNN reaches to reference matrix (Figure 8(b)) that correctly presents the nature of noise. Because the EKF operates with fixed matrix R so that the deviations between the true path and the estimated by EKF are larger than by FNN-EKF in x, y, θ direction respectively (Figure 8(c), (d), (e)).

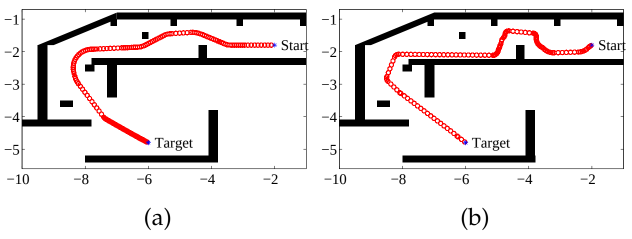
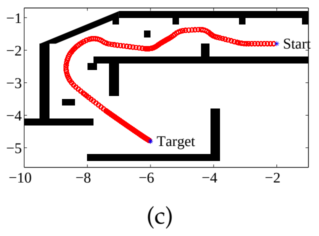



Figure 9: Travelling paths of the robot navigated by three modules: (a) BBFM, (b) MOASM, (c) CDB.

3.2 Comparing the BBFM to other modules

The position of robot is direct feedback and the noises are not considered in this assessment. The MOASM uses multi-objective optimization for command fusion. It is implemented with three objective functions including avoiding obstacles, maintaining the target heading and moving fast forward. Three objective functions are equal two behaviors in the BBFM and defined as in the origin [17]. The overall control signal is determined by using the lexicographic method. The CDB is implemented with two behaviors as in the BBFM. Each behavior is a fuzzy controller. The overall control signal is determined by using fuzzy-meta rules and defuzzification.

In this case, the operating environment is chosen to be the same as in the original paper of MOASM [17]. The start position is (-2, -1.8, 180^0) and the target position is (-6, -4.8, 0^0). Figure 9 shows the path of robot generated by three various modules MOASM, BBFM, and CDB. It can be seen that all modules successfully navigate the robot to avoid obstacles and reach the target. Table III shows the average performance of each module, where the index smoothness is the average absolute value of the difference between the current and the previous orientation, thus showing how smooth the maneuvers is; the index target error is the distance between the final position of the robot and the target position, thus evaluating the reachable ability to the target at the steady state; and the indexes traveled distance and elapsed time are respectively the total distance of the robot's traveled path and the time taken to go through that path. As can be inferred from Table III, the BBFM is more efficient than the remaining architectures in almost all criteria.

3.3 Combining the BBFM with FNN-EKF

The operating environment is chosen to be more like an office with wall and bulkhead obstacles in this case. The start position is (-7, -6, 0^0) and the target is (-2.5, -1.5, 0^0). In order to evaluate the efficiency of FNN-EKF

Table III: Navigation Results in Case 1

Index	BBFM	MOASM	CDB
Traveled distance (m)	10.36	11.02	11.02
Elapsed time (second)	28.26	41,45	36.43
Smoothness (degree)	0.88	1.29	6.1
Target error (m)	0.05	0.2	0.05

Table IV: Navigation Results in Case 2

Index	BBFM&FNN-EKF	BBFM
Traveled distance (m)	9.1	23.9
Smoothness (degree)	2.36	8.93
Target error (m)	0.05	0.05

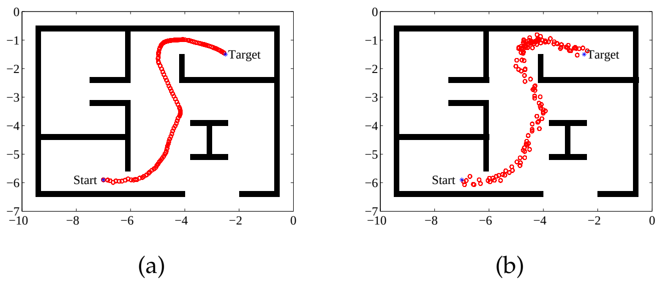


Figure 10: Traveling path of the robot navigated by: (a) BBFM&FNN-EKF, (b) BBFM.

when combining in the navigation system, the comparison of the position of robot is shown in two scenarios: navigation system uses only BBFM and in case uses both BBFM and FNN-EFK as shown in Figure 10. The Figure 11(a), (c), and (e) are comparisons the true position, the position using only BBFM, and positon using BBFM&FNN-EKF, while (b), (d), and (f) are the error between the true path and the path using BBFM and the true path and the path using BBFM&FNN-EKF in x, y, and heading direction respectively. Table IV shows the performance of two scenarios such as traveled path, smoothness, and target error. It can be seen from these results that navigation system using BBFM & FNN-EFK is more efficient than that using only BBFM in unknown and noise affected navigation system.

4 Experiments

4.1 Experimental Setup

The robot used in experiments is a Sputnik robot [19] as shown in Figure 12. It is equipped with three ultrasonic sensors DUR5200 at left, front and right directions creating the scanning range from -60^0 to 60^0. To extend the scanning range to $[-90^0, 90^0]$, we added two additional ultrasonic sensors SRF05 to the left and right sides of the robot. Each employs a micro-controller PIC12F1572 to synchronize its data with the main board of Sputnik robot. The linear and angular velocities of the robot are respectively set to [0, 0.5] m/s and [-3.7, 3.7] rad/s. The position of the robot is determined via optical encoder sensors and the

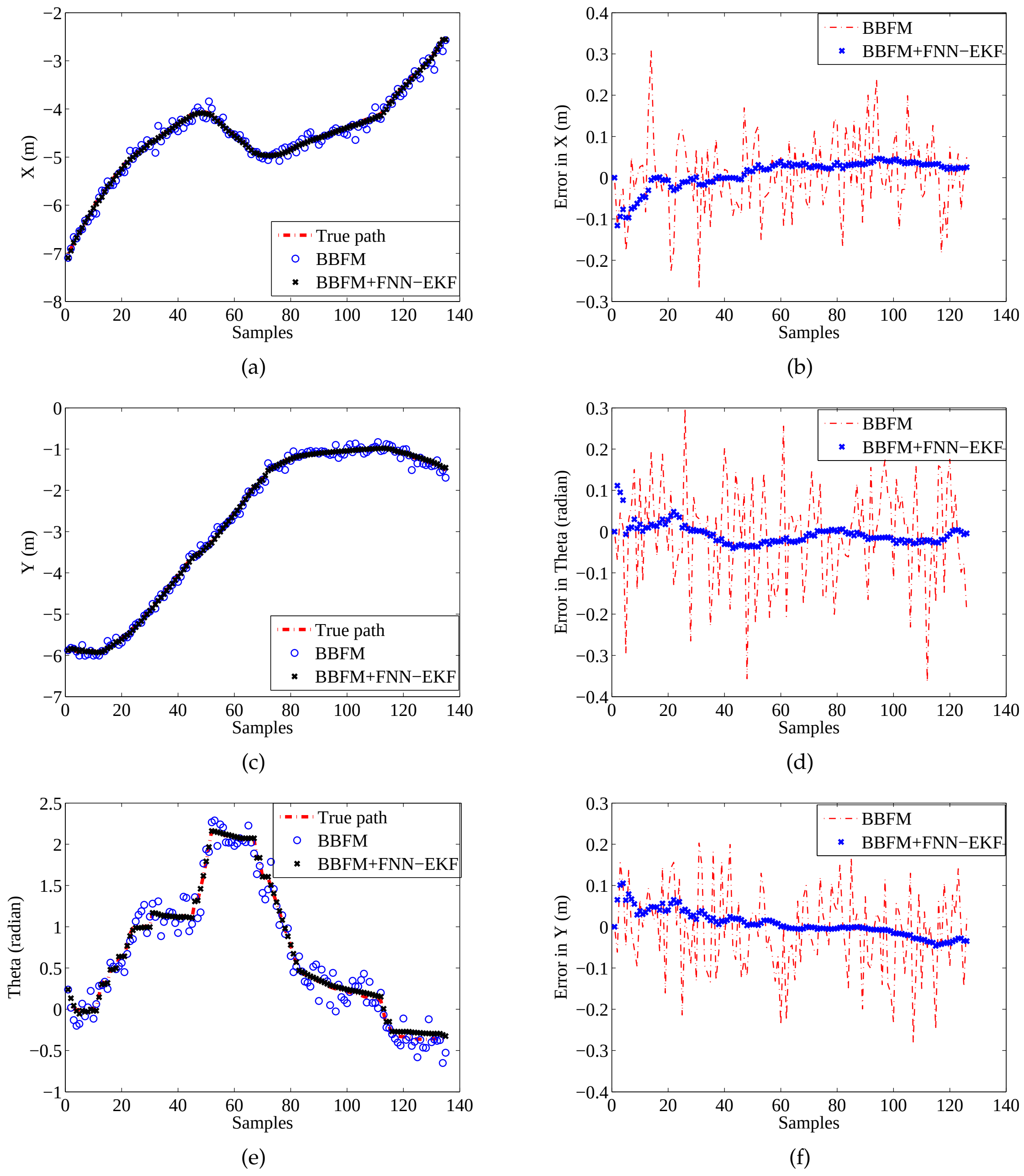


Figure 11: The comparison of the position of robot using BBFM and BBFM & FNN-EFK in: (a) and (b): x-axis, (c) and (d): y-axis, (e) and (f): the orientation

orientation is determined by compass sensor. The robot has a wireless module connecting it to a Wifi router (Figure 12). The navigation system with BBFM module and the FNN-EKF module is written in Matlab and installed on a PC which communicates with the robot through a Wifi router. The experimental environment is an indoor office with the size of 4 m × 3 m and changeable obstacles.

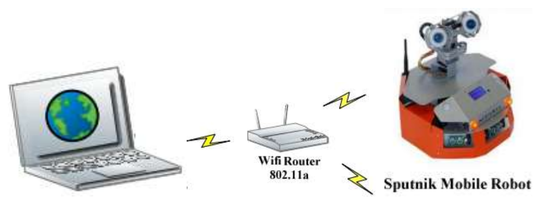


Figure 12: The Sputnik robot and its configuration to communicate with the control computer.

4.2 Experimental Results

Experiments were carried out with different configurations of the environment and target position. In Case 1 (Figure 13(a)), the robot starts at A (0, 0, 90^0) and turns right following to direction of the target. The robot does not change the direction from A to B to avoid obstacles. At B, it turns left following to direction of the target again to C. At C, the robot continuously adjusts its direction and then goes straight along wall to the target D (1.5, 3, 0^0). Figure 13(b) shows the correspondence of linear and angular velocities of the robot with those

Table V: Navigation Results in Two Cases

Case	Smoothness (degree)	Elapsed time (s)	Traveled distance (m)	Target Error (m)
1	1.36	25.5	3.48	0.04
2	4.3	30	4.29	0.06

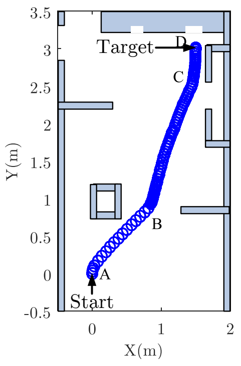
movements, for instance, at B, the angular velocity switches between the left and right directions to get the target while the linear velocity gradually decreases near to the target. There are also no changing of the angular velocity when the robot is moving forward. The images of the robot during operating are shown in Figure 13(c).

In Case 2 as shown in Figure 13, the environment structure changed with more obstacles. The robot starts at position (0, 0, 180^0) and the target is set to (-1.1, 3, 0^0). The path from A goes along wall to B and turns right to C. From C to D shows that the robot successfully escapes obstacles two sequential times and goes to E. Then the robot turns left and goes forward to the target. Figure 13(e) shows the correspondence of the robot's velocity with its movements. Figure 13(f) are some operating images of robot.

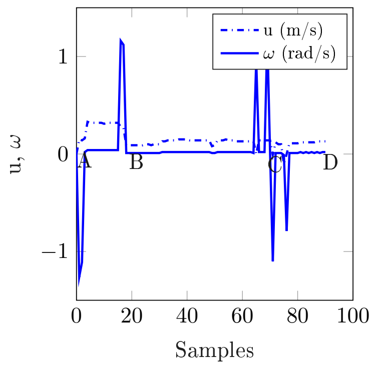
Table V shows the navigation performance in each case. As can be inferred from the table, Case 1 introduces the best performance in smoothness ad elapsed time because it is the simplest case. On the contrary, Case 2 has worse performance due to obstacles. Nevertheless, the closeness between values of average linear velocities determined by the ratio of traveling distance and elapsed time implies that the operation of robot is stable and suitable for indoor environment.

5 Conclusions

In this paper, we have proposed a navigation system for the mobile robot in unknown environments. The performance of navigation system is improved by using two effective main modules including the FNN-EKFn module for localization and the BBFM module for navigation. The FNN-EKF employs a neural-fuzzy system to regulate the noise covariance matrix so that the estimatiton is converged and more accurate. The good localization result is used as observation data for the BBFM. The BBFM inherits advantages of fuzzy logic in dealing with uncertainties of sensor information and also takes advantage of multi-objective optimization to generate Pareto-optimal solutions for command fusion. The results show that the proposed system possibly navigates the robot to reach the target along an efficient trajectory in environments with unpredictable obstacles and noises.

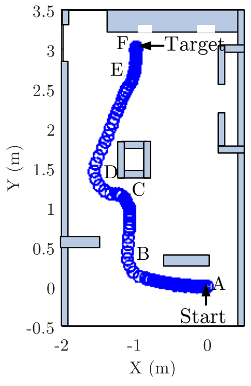


(a)

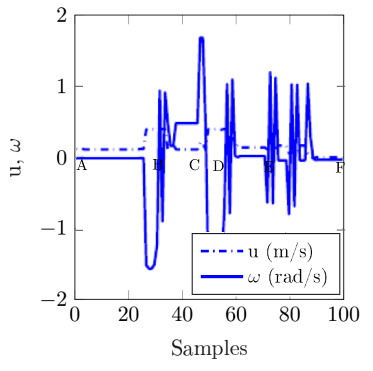


(b)

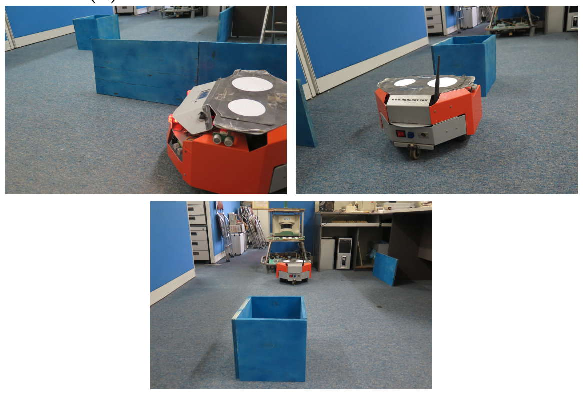
(c)



(d)



(e)



(f)

Figure 13: Travelling path, velocity response and real images of the robot navigated in: Case 1: (a)-(c), Case 2 (e)-(f).

References

[1] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*. The MIT Press Cambridge, 2004.

[2] J. Borenstein, H. R. Everett, L. Feng, and D. K. Wehe, "Mobile robot positioning: Sensors and techniques,"

Journal of Robotic Systems, vol. 14, no. 4, pp. 231–249, 1997.
[3] G. Bishop and G. Welch, "An introduction to the kalman filter," *Proc of SIGGRAPH, Course*, vol. 8, 2001.
[4] V. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello, "Bayesian filtering for location estimation," *IEEE Pervasive Computing*, vol. 2, no. 3, pp. 24–33, 2003.
[5] D. Crisan and A. Doucet, "A survey of convergence results on particle filtering methods for practitioners," *IEEE Transactions on Signal Processing*, vol. 50, no. 3, pp. 736–746, 2002.
[6] D. Nakhaeinia, S. Tang, S. M. Noor, and O. Motlagh, "A review of control architectures for autonomous navigation of mobile robots," *International Journal of Physical Sciences*, vol. 6, no. 2, pp. 169–174, 2011.
[7] M. Dorigo and M. Colombetti, *Robot shaping: an experiment in behavior engineering*. MIT Press, 1998.
[8] R. C. Arkin, "Motor schema-based mobile robot navigation," *The International journal of Robotics Research*, vol. 8, no. 4, pp. 92–112, 1989.
[9] E. Freire, T. Bastos-Filho, M. Sarcinelli-Filho, and R. Carelli, "A new mobile robot control approach via fusion of control signals," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 1, pp. 419–429, 2004.
[10] A. Adriansyah and S. Amini, "Genetic fuzzy system in behavior based mobile robot," *Politeknik Elektronika Negeri Surabaya ITS*, 2004.
[11] A. S. Al Yahmedi and M. A. Fatmi, "Fuzzy logic based navigation of mobile robots," in *Recent Advances in Mobile Robotics*. InTech, Dec. 2011, iSBN: 978-953-307-909-7, DOI:10.5772/25621.
[12] H. Mo, Q. Tang, and L. Meng, "Behavior-based fuzzy control for mobile robot navigation," *Mathematical problems in engineering*, vol. 2013, 2013.
[13] A. AlYahmedi, E. El-Tahir, and T. Pervez, "Behavior based control of a robotic based navigation aid for the blind," in *inproceedings of the Control & Applications Conference (CA2009)*, 2009.
[14] M. Faisal, K. Al-Mutib, R. Hedjar, H. Mathkour, M. Alsulaiman, and E. Mattar, "Behavior based mobile for mobile robots navigation and obstacle avoidance," *International Journal of Computers and Communications*, vol. 8, 2014.
[15] D. Driankov, H. Hellendoorn, and M. Reinfrank, *An Introduction to Fuzzy Control*. Springer, 2010, iSBN: 978-3-642-08234-4.
[16] S. Naaz, A. Alam, and R. Biswas, "Effect of different defuzzification methods in a fuzzy based load balancing application," *International Journal of Computer Science Issues*, vol. 8, no. 1, pp. 261–267, 2011.
[17] P. Pirjanian, "Multiple objective behavior-based control," *Robotics and Autonomous Systems*, vol. 31, no. 1, pp. 53–60, 2000.
[18] L. Teslić, I. Škrjanc, and G. Klančar, "Ekf-based localization of a wheeled mobile robot in structured environments," *Journal of Intelligent & Robotic Systems*, vol. 62, no. 2, pp. 187–203, 2011.
[19] Dr Robot Manual. [Online]. Available: http://www.drrobot.com/products

Thi Thanh Van Nguyen received B.Sc degree in Electronics and Telecommunication from Vietnam National University, Hanoi in 2001. She worked at VNU University of Engineering and Technology as a graduate trainee (2001-2004). She gained M.S degree in Mechatronics from Asia Institute of Technology, Thailand in 2006. She has been a lecturer at the Faculty of Electronics and Telecommunications (FET), VNU University of Engineering and Technology, since 2007. Her research and teaching interests are electronics engineering, control engineering, localization and navigation for autonomous mobile robot system.

Ha Vu Le received a B.E. degree in Informatics from Hanoi University of Technology, Vietnam in 1993, an M.Sc. degree and a Ph.D. degree, both in Computer Science, respectively from the California State Polytechnic University in 2000 and from the Center on Advanced Computer Studies, University of Louisiana at Lafayette, USA in 2003. He has been a lecturer at the Faculty of Electronics and Telecommunications (FET), VNU University of Engineering and Technology, since 2004. He worked as a researcher at the Institute of Information Technology, National Academy of Science and Technology (1993-1997), and as a post-doctoral researcher at the Institute of Fundamental Electronics, University of Paris-Sud 11 (2006-2007). He is currently Head of the Signals and Systems Laboratory at FET. His research and teaching interests include machine vision and 3D vision, automatic image and video annotation, super-resolution imaging, bio-medical imaging and signal processing, and multimedia communications. Dr. Le is member of the IEEE Computer Society and Signal Processing Society (USA), IEICE (Japan), and REV (Vietnam). He has served as an editorial board member of the REV Journal on Electronics and Communications and as an organizing committee member of the International Conferences on Advanced Technologies for Communications (2008-2012). He has also served as technical committee member and reviewer for several international journals and conferences, including Digital Signal Processing journal, EURASIP Journal on Embedded Systems, and EURASIP Journal on Advances in Signal Processing.

Quang Vinh Tran defended a doctoral degree at Vietnam National University (VNU) basing on experiments conducted in the Technical University of Vienna (Austria). He now is Associate Professor, Head of Electronics and Computer Engineering Department, VNU. His research interests include computer architecture, measurement and control using computer, intelligent robotics, autonomous mobile and networked robotics.