Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 9315925, 11 pages
https://doi.org/10.1155/2018/9315925

Research Article

A Multiagent Architecture for Mobile Robot Navigation Using Hierarchical Fuzzy and Sliding Mode Controllers

Dorra Ayedi,[1] **Maïssa Boujelben**,[2] **and Chokri Rekik**[2]

[1] *Control and Energy Management Lab (CEM Lab), Sousse Engineering School, University of Sousse, BP 264, Sousse Erriadh, 4023 Sousse, Tunisia*
[2] *Control and Energy Management Lab (CEM Lab), Sfax Engineering School, University of Sfax, BP W, 3038 Sfax, Tunisia*

Correspondence should be addressed to Dorra Ayedi; dorra.ayedii@yahoo.fr

Received 29 September 2017; Revised 18 December 2017; Accepted 16 January 2018; Published 13 February 2018

Academic Editor: Rafael Morales

The challenge of this work is to implement an algorithm which enables the robot to achieve independent activities in the purpose of achieving a common goal, which consists in autonomous navigation in a partially unknown environment. The use of multiagent system is convenient for such a problem. Hence, we have designed a structure composed of four agents dedicated to perception, navigation, static, and dynamic obstacle avoidance. Those agents interact through a coordination system.

1. Introduction

In recent years, multiagent system becomes a very well-known field used in many applications such as E-commerce, E-health applications, network intrusion detection systems, telematic and transport systems [1], and robotic system [2, 3], which is the topic of this paper.

Indeed, several researchers have used multiagent system in the development of robotic system [4]. Multiagent system is very useful in the case of multirobots which execute different tasks, which should be coordinated with each other [5].

Furthermore, such architecture can be used in the case of a single robot. According to the principle of the multiagent system, which consists in a number of agents coordinated and dedicated to achieve common objectives, each robot activity is considered as an agent. Those agents are interacting with each other through a specific system of coordination.

The interest in the multiagent system has increased, since it has several properties compared to other architectures [6]. Among those properties, we mention decentralized control, which permits the continuity of the system even when some parts fail, coordinability, predictability, and adaptability [2, 3].

Therefore, the structure of multiagent system may be established as a first step. Then, the activity of each agent is defined. Finally, a coordination system between agents is presented. Those steps are applied to implement the multiagent system on an intelligent wheelchair in [7]. Five agents were designed by Busquets et al. for robot navigation [8]. Innocenti et al. have proposed in [2] a multiagent structure containing four agents, combined with each other in order to obtain the desired goal. Boujelben et al. have simplified this architecture and reduced the number of agents to three, dedicated to perception, navigation, and obstacle avoidance [3].

Among the agents defined in the case of robotic field, we cite navigation agent and obstacle avoidance agent. In the literature, the design of controllers for mobile robot navigation is based on two basic approaches: the reactive and the deliberative approach [9].

The deliberative method is applied when the situation needs a previous knowledge of the environment [10]. Path planning is the principle of this approach. Researchers have developed methods of control based on deliberative approach such as PID control [11] and sliding mode [12].

For an unknown environment, the deliberative approach is not sufficient. Therefore, the reactive method is required, which is based on sensor's measurements and does not require information about the environment [9]. In the literature, neural network [13], potential field [14], and

fuzzy logic system [15] are the most popular developed methods.

In this work, we have treated the problem of an environment containing both static and dynamic obstacles. Our mobile robot should find a safe path to reach the desired target. For that purpose, we have proposed to apply the multiagent system to control the robot. Our suggested structure contains four agents:

(i) Perception agent, which obtains the necessary information measured by the sensors of the robot and affords them to the system

(ii) Navigation agent, responsible for guiding the robot to the target

(iii) Static obstacle avoidance, which keeps the robot away from static obstacles

(iv) Dynamic obstacle avoidance, which keeps the robot away from dynamic obstacles

Those agents are related through a cooperative system which allows them to achieve common goals.

Here, the navigation agent uses a simple fuzzy controller. Indeed, Chia-Feng Juang has tested the efficiency of fuzzy logic system in the field of robot navigation [16], comparing the performance of this method, while introducing some optimization approaches. In this work, fuzzy logic system was implemented in the navigation agent for its high robustness and good performances, without applying optimization methods.

For the static obstacle avoidance agents, a deliberative method is applied, since the position of obstacles is known. Sliding mode is chosen to be implemented on this agent thanks to its fast response and its robustness against variations. For the dynamic obstacles, the position of obstacles is unknown and unexpected. The deliberative approach is not sufficient, due to the complicated computation of the following path in the presence of dynamic obstacle. Fuzzy logic system is applied, since it is considered as a reactive method.

However, a large number of inputs are obtained in the case of applying the standard fuzzy logic system, due to the high quantity of information given by all robot's sensors. To simplify the problem, we have proposed to use the hierarchical fuzzy system which consists in dividing the fuzzy system into subsystems [15].

This paper is organized as follows. In Section 2, the model of the used robot is presented. Section 3 introduces the adopted multiagent architecture, detailing the role of each agent. Coordination between agents is mentioned in Section 4. Finally, Section 5 presents the simulation results to test the validity of this work.

2. Modeling of the Mobile Robot

The platform used in this work is a Khepera II robot. This mobile robot is convenient for our experiments, due to its equipment consisting on two independent wheels and eight sensors attached to its contour which ensure a best detection to the obstacles, as it is shown in Figure 1 [17].

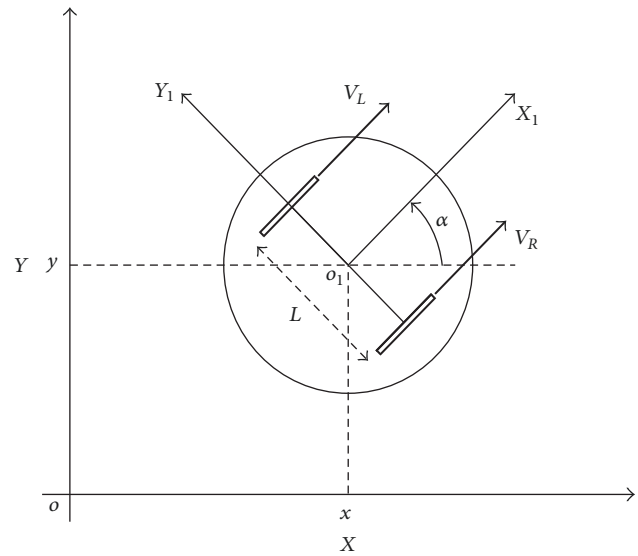

Figure 1: The schematic model of the Khepera II robot.

The linear right and left wheels velocities V_R and V_L as well as the angle α between the robot direction and the x-axis are responsible for guiding the robot, using the following kinematic model:

$$\begin{aligned} \dot{x} &= \frac{V_R + V_L}{2} \cos\alpha, \\ \dot{y} &= \frac{V_R + V_L}{2} \sin\alpha, \\ \dot{\alpha} &= \frac{V_R - V_L}{L}, \end{aligned} \tag{1}$$

where L is the distance between the right and the left wheels.

The adopted strategy is to control the robot to a final destination in an environment containing both static and dynamic obstacles using the multiagent system.

3. Multiagent Architecture Model

The basic structure of the multiagent model is presented in Figure 2.

It is composed of four agents which interact with each other through a cooperative system. This system of coordination will be detailed in Section 5.

The information provided by the sensor's measurements is collected in the perception agent. According to those data, one of the three other agents (robot navigation, static obstacle avoidance, and dynamic obstacle avoidance) is chosen to control the robot. If the environment is safe, the robot navigation agent is activated, which is responsible for guiding the robot from an initial position to a final target. If an obstacle is located on the path of the robot, one of the obstacle avoidance agents is activated. Those two agents are in charge of moving away from obstacles, one of them applying a reactive method and the second using a deliberative approach [18]. The choice between those two agents depends on type of obstacle faced (static, dynamic).

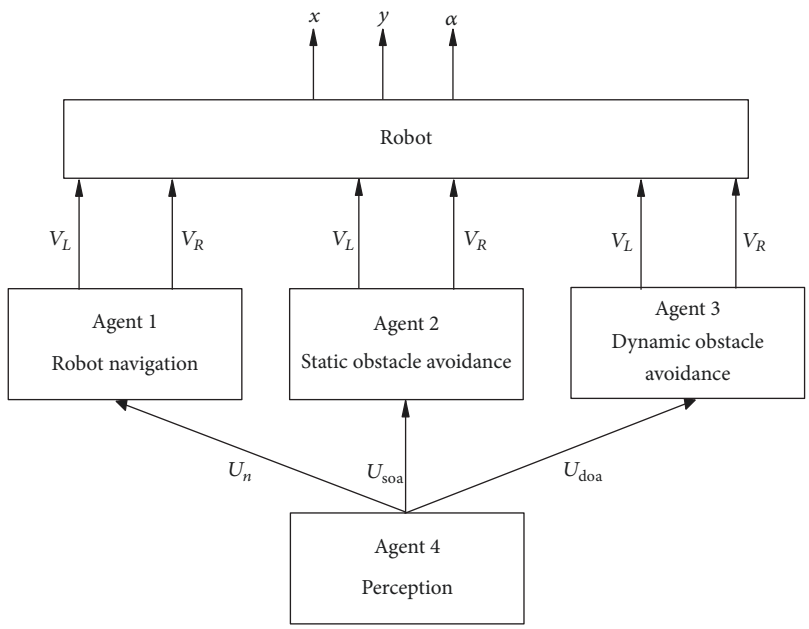

Figure 2: The multiagent system architecture.

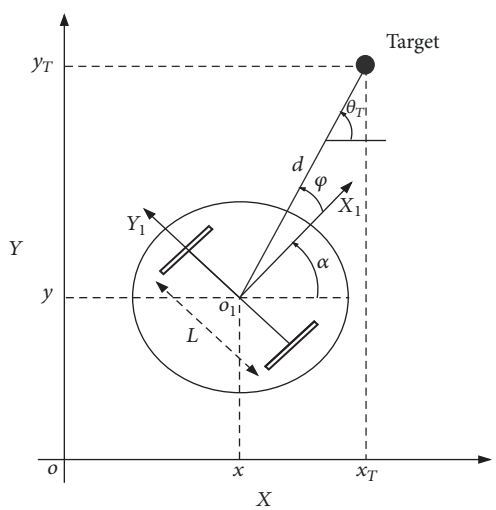

Figure 3: Representation of the parameters d and φ.

3.1. Navigation Agent. The role of the navigation agent, based on fuzzy logic control, is to bring the robot to a final target when the environment is supposed to be free from obstacles [3]. This task is achieved by calculating the distance d separating the robot center from the goal and the angle φ which separates the distance to the target and the robot orientation (see Figure 3), then giving them as inputs to a fuzzy logic controller [19]. This controller is responsible for the function of navigation.

The distance d and the angle φ are calculated using those following expressions:

$$d = \sqrt{(x_T - x)^2 + (y_T - y)^2}, \tag{2}$$

$$\varphi = \theta_T - \alpha_R \tag{3}$$

with

$$\theta_T = \arctan \frac{(y_T - y)}{(x_T - x)}. \tag{4}$$

Fuzzy logic system is a well-known technique recommended for robust controller's conception [20]. Its advantage consists in considering the variation field of input variables, unlike Boolean logic in which the values of the variables are 0 or 1 [3, 21].

Figures 4 and 5 show the fuzzy partition of the input variables d and φ. The distance d varies in the range $[0, 700]$ mm, whereas the interval of the angle φ is defined between $-\pi/2$ and $\pi/2$. Five membership functions are used for the distance, and seven memberships functions are considered for the angle. Therefore, 35 rules are obtained. More details and inference tables are provided in [15, 21].

An example of a fuzzy rule which represents a link between fuzzy inputs variables (d and φ) and fuzzy outputs variables (V_L and V_R) is expressed as follows.

If d is A_i and φ is B_i, then $V_R = y_i$ and $V_L = z_i$, where α_i and β_i are the level activation of the rule i.

Therefore, the expressions of the output variables are

$$V_R = \frac{\sum_{i=1}^{r} \alpha_i y_i}{\sum_{j=1}^{r} \alpha_j}, \qquad V_L = \frac{\sum_{i=1}^{r} \beta_i z_i}{\sum_{j=1}^{r} \beta_j}. \tag{5}$$

3.2. Static Obstacle Avoiding Agent. Now, we study the case of the existence of a static obstacle in the path of the robot. To avoid it, a deliberative approach is applied, which consists in defining a trajectory that the robot should follow [22, 23]. The proposed approach is based on following a specific trajectory

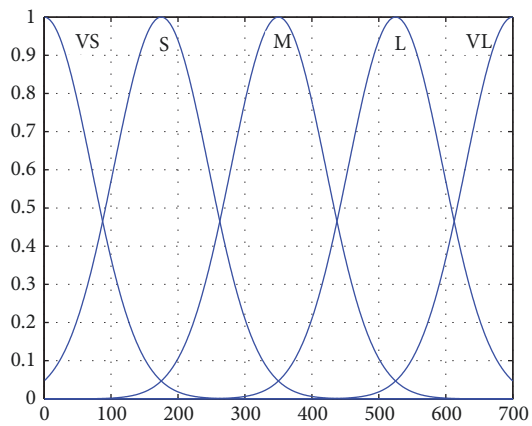

Figure 4: Membership functions for d (in mm).

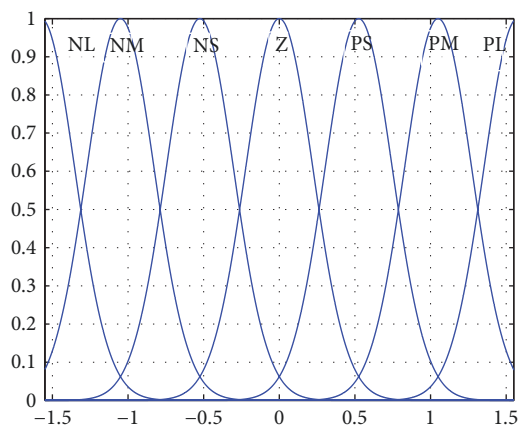

Figure 5: Membership functions for φ (in rad).

generated by limit cycle equations applying the sliding mode control [22]. Some details about the limit cycle trajectory and the sliding mode control are presented in the following subsections.

3.2.1. Limit Cycle Trajectory. First of all, we define a convergence circle around each static obstacle. According to previous works [22, 24], this circle corresponds to a dangerous zone. Its radius is defined as $R_c = R_r + R_{os} + \delta$ (see Figure 6), where

(i) d is the distance between the robot center and the target,

(ii) d_{os} is the distance between the robot center and the static obstacle,

(iii) $R_r$is robot's radius,

(iv) $R_{os}$is static obstacle's radius,

(v) δ is a safety margin for collision avoidance.

Now we consider the following differential equation, which defines the limit cycle trajectory of the robot [25]. This system of equation permits us to obtain the position (x_r, y_r)

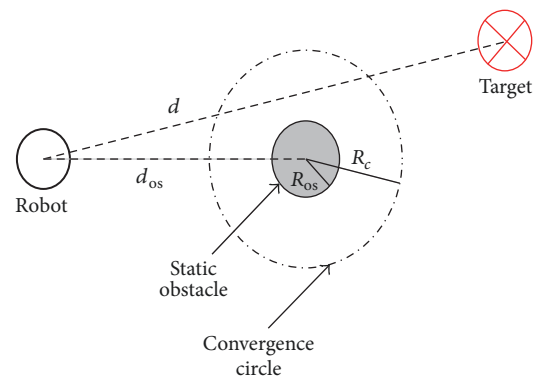

Figure 6: The convergence circle.

of the robot enabling it to track the convergence cycle defined previously [22]:

$$\begin{aligned} \dot{x}_r &= a y_r + x_r \left(R_c^2 - x_r^2 - y_{r^2}\right), \\ \dot{y}_r &= -a x_r + y_r \left(R_c^2 - x_r^2 - y_{r^2}\right), \end{aligned} \tag{6}$$

where a is a scalar which specifies the orbital path direction (clockwise or counterclockwise), according to the position of the robot with respect to the obstacle. If $a = 1$, the rotational direction is clockwise. If $a = -1$, this direction is counterclockwise. Figure 7 shows both cases with different initial conditions.

In the purpose of determining the appropriate direction, a new reference frame is defined $S = (C, X_o, Y_o)$ centered on the obstacle, as it is presented in Figure 8.

Applying this reference, the new coordinates of the robot are obtained (x_{or}, y_{or}) [24]. Then the sign of y_{or} determines the motion direction (clockwise if $y_{or} > 0$ and counterclockwise otherwise).

3.2.2. Sliding Mode Control. Once the reference trajectory has been determined, the approach proposed for tracking this path is the sliding mode control. This method ensures fast response, robustness, and good results in tracking the desired trajectory [26].

This method is based on choosing the appropriate sliding surface denoted by s along which the sliding motion occurs [12, 26].

To illustrate the principle of the sliding mode control, we consider Figure 9, where the tracking error $p_e = (x_e, y_e, \alpha_e)^T$ between the reference portion $p_r = (x_r, y_r, \alpha_r)^T$ and the current position $p = (x, y, \alpha)^T$ of the robot is represented and expressed by

$$\begin{bmatrix} x_e \\ y_e \\ \alpha_e \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_r - x \\ y_r - y \\ \alpha_r - \alpha \end{bmatrix}. \tag{7}$$

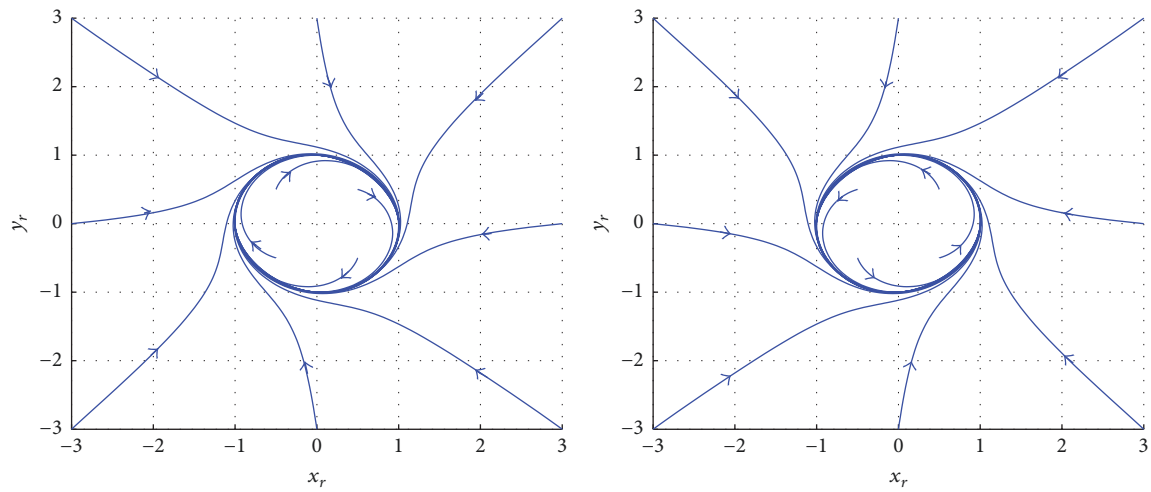

Figure 7: The shape of the limit cycles, (a) clockwise and (b) counterclockwise.

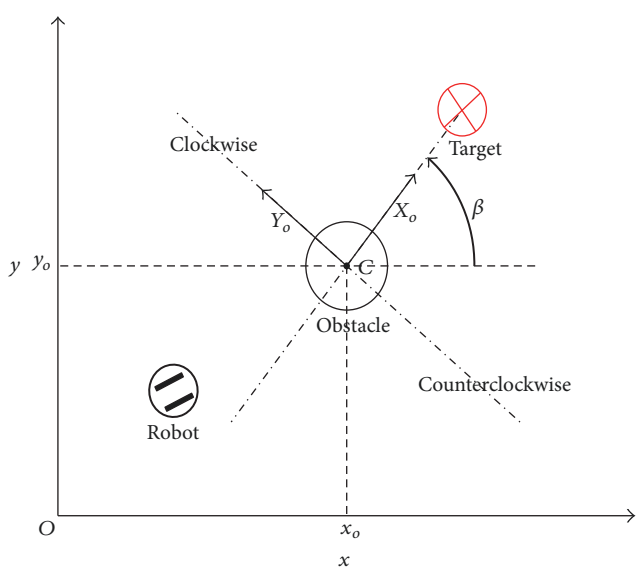

Figure 8: The new reference frame $S = (C, X_o, Y_o)$.

By deriving this system of equations, we obtain

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\alpha}_e \end{bmatrix} = \begin{bmatrix} y_e w - v + v_r \cos \alpha_e \\ -x_e w + v_r \sin \alpha_e \\ w_r - w \end{bmatrix}. \tag{8}$$

The error portion p_e must converge to 0, which means that p converges to p_r. To achieve this purpose, the adequate linear and angular velocities v and w should be found.

The design of the switching function is a hard task due to the robot model which is a multiple-input nonlinear system [22]. Lee et al. have proposed in [12] to consider $x_e = 0$. As a consequence, the Lyapunov equation is chosen as

$$V = \frac{1}{2} y_e^2. \tag{9}$$

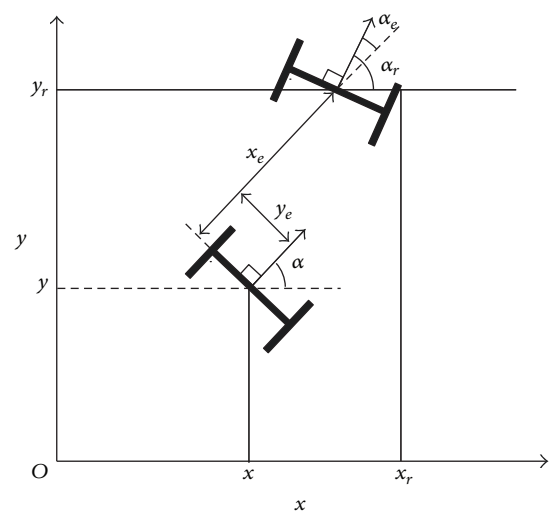

Figure 9: Tracking error.

By deriving V, we obtain

$$\begin{aligned} \dot{V} &= y_e \dot{y}_e = y_e \left(-x_e w + v_r \sin(\alpha_e)\right) \\ &= -x_e y_e w - v_r y_e \sin\left(\arctan\left(v_r y_e\right)\right), \end{aligned} \tag{10}$$

where $\alpha_e = -\arctan(v_r y_e)$ is the switching function condition.

α_e is always positive. Therefore, the global reaching condition given as $\dot{V} \leq 0$ is always satisfied [22]. Then we define the vector of sliding surfaces as

$$s = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} x_e \\ \alpha_e + \arctan\left(v_r y_e\right) \end{bmatrix}. \tag{11}$$

If we force s to converge to 0, consequently $x_e \to 0$ and $\alpha_e \to -\arctan(v_r y_e)$, which leads to $y_e \to 0$.

To reduce the problem of chattering [12], we replace the switching function by a saturation function defined as [27]

$$\dot{s} = -k \operatorname{sat}(s). \tag{12}$$

By substituting $\arctan(v_t y_e)$ with γ, we obtain

$$\begin{aligned}\dot{s} &= \begin{bmatrix} \dot{s}_1 \\ \dot{s}_2 \end{bmatrix} = \begin{bmatrix} -k_1 \operatorname{sat}(s_1) \\ -k_2 \operatorname{sat}(s_2) \end{bmatrix} \begin{bmatrix} \dot{x}_e \\ \dot{\alpha}_e + \dfrac{\partial \gamma}{\partial v_r}\dot{v}_r + \dfrac{\partial \gamma}{\partial y_e}\dot{y}_e \end{bmatrix} \\ &= \begin{bmatrix} y_e w - v + v_r \cos\alpha_e \\ w_r - w + \dfrac{\partial \gamma}{\partial v_r}\dot{v}_r + \dfrac{\partial \gamma}{\partial y_e}\left(-x_e w + v_r \sin\alpha_e\right) \end{bmatrix}.\end{aligned} \tag{13}$$

The obtained control law is

$$\begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} y_e w + v_r \cos\alpha_e + k_1 \operatorname{sat}(s_1) \\ w_r + \dfrac{\partial \gamma}{\partial v_r}\dot{v}_r + \dfrac{\partial \gamma}{\partial y_e}\left(v_r \sin\alpha_e\right) + k_2 \operatorname{sat}(s_2) \end{bmatrix} \tag{14}$$

with $\partial\gamma/\partial v_r = y_e/(1+(v_r y_e)^2)$ and $\partial\gamma/\partial y_e = v_r/(1+(v_r y_e)^2)$.

3.3. Dynamic Obstacle Avoiding Agent. In the following section, we consider that the robot faces on his path dynamic obstacles. For its safety, the robot must know at every instance the position of the obstacle to avoid it [15]. The idea is to calculate at each moment the distance d expressed in (2). If this distance is lower than 50 mm, we apply the hierarchical fuzzy system as a reactive approach for obstacle avoidance.

We have chosen to work with the hierarchical fuzzy system rather than the standard fuzzy system due to the huge number of rules obtained if we apply one fuzzy controller [17, 21]. The principle of this method consists in decomposing the fuzzy system into subsystems having lower size and related to each other in a hierarchical way. This structure is presented in Figure 10.

According to this block diagram, each controller from the first layer is related to a sensor of the Khepera II robot. This sensor is required to send the distance d_i and the angle φ_i ($i = 1, \ldots, 8$) to an ith controller, which gives as inputs the left and right wheels velocities V_{Li} and V_{Ri}, and an index denoted by I_0^i. This index indicates the degree of collision between the obstacle and the robot [21, 28]. This degree is expressed as follows:

$$I_0^i = \frac{\sum_{j=1}^{35} \mu_j(d_i, \varphi_i) I_j}{\sum_{j=k}^{35} \mu_k(d_i, \varphi_i)}. \tag{15}$$

The expressions of V_{Li} and V_{Ri} are given as

$$\begin{aligned} V_{Ri} &= \frac{\sum_{j=1}^{r} \mu_j(d_i, \varphi_i) y_j}{\sum_{k=1}^{r} \mu_k(d_i, \varphi_i)}, \\ V_{Li} &= \frac{\sum_{j=1}^{r} \mu_j(d_i, \varphi_i) z_j}{\sum_{k=1}^{r} \mu_k(d_i, \varphi_i)}, \end{aligned} \tag{16}$$

where I_j is the consequence of the jth rule and μ_j is the activation level of the jth rule.

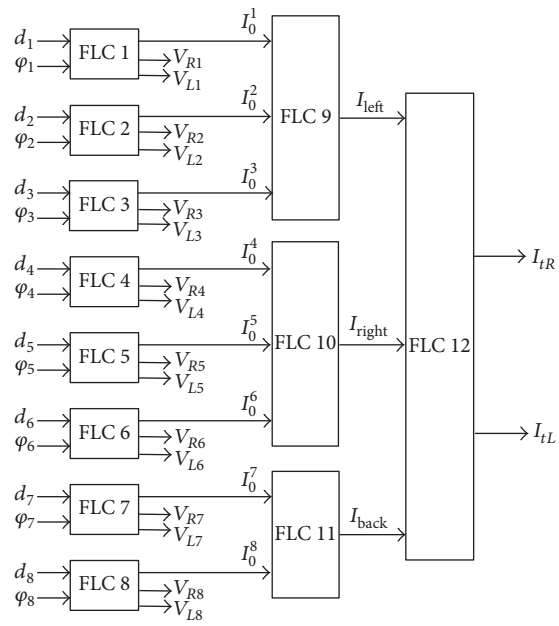

Figure 10: Block diagram of the hierarchical fuzzy system.

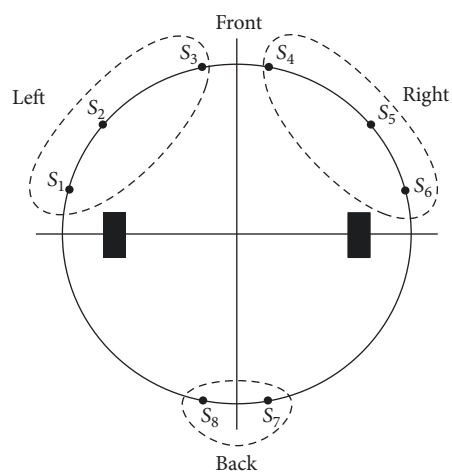

Figure 11: Position of the eight sensors of the robot.

Now the controllers are regrouped into three groups as the eight sensors of the Khepera II robot (see Figure 11). This decomposition is made in order to detect the obstacles existing on the right, left, and back sides.

The controllers related to each group of sensors are associated with a fuzzy controller. Therefore, we obtain three controllers which give the indexes I_{left}, I_{right}, and I_{back}. Those new indexes indicate the collision degree between the robot and the obstacle existing on the left, the right, and back sides, respectively.

Finally, I_{left}, I_{right}, and I_{back} are associated with a last fuzzy controller. The outputs of this controller are I_{tR} and I_{tL}. Unlike the previous indexes, those two last indicators give information about the absence of obstacle around the robot. I_{tR} and I_{tL} are expressed as follows:

$$I_{tR} = \frac{\sum_{j=1}^{3}\sum_{i=1}^{3}\mu_{ij}I_{ij}^{R}}{\sum_{l=1}^{3}\sum_{k=1}^{3}\mu_{kl}},$$
$$I_{tL} = \frac{\sum_{j=1}^{3}\sum_{i=1}^{3}\mu_{ij}I_{ij}^{L}}{\sum_{l=1}^{3}\sum_{k=1}^{3}\mu_{kl}}. \quad (17)$$

The inference tables of all the indexes mentioned previously are given in [3, 28].

At least, the velocities expressions are written as

$$V_R = I_{tR}V_{tR} + \sum_{i=1}^{n} I_0^i V_{Li},$$
$$V_L = I_{tL}V_{tL} + \sum_{i=1}^{n} I_0^i V_{Ri} \quad (18)$$

with

$$V_{Ri} = \frac{\sum_{j=1}^{r}\mu_j(d_i,\varphi_i)\,y_j}{\sum_{k=1}^{r}\mu_k(d_i,\varphi_i)},$$
$$V_{Li} = \frac{\sum_{j=1}^{r}\mu_j(d_i,\varphi_i)\,z_j}{\sum_{k=1}^{r}\mu_k(d_i,\varphi_i)},$$
$$V_{tR} = \frac{\sum_{j=1}^{r}\mu_j(d,\varphi)\,y_j}{\sum_{k=1}^{r}\mu_k(d,\varphi)},$$
$$V_{tL} = \frac{\sum_{j=1}^{r}\mu_j(d,\varphi)\,z_j}{\sum_{k=1}^{r}\mu_k(d,\varphi)}. \quad (19)$$

4. Cooperative System between Agents: Utility Function

On this section, the cooperative control system is detailed. In fact, the coordination between the agents is necessary to avoid contradictory actions [2].

For this purpose, the utility function is proposed [2, 3]. The idea consists in giving each agent an utility function. According to the value obtained from this function, which varies between 0 and 1, the choice of the agent which controls the robot is made.

We begin by introducing the navigation utility function U_n presented in Figure 12.

This function is drawn according to the system defined as below:

$$U_n = \begin{cases} 1 & \text{if } d \le 100 \\ \alpha * d + \beta & \text{if } 100 < d < 500 \\ 0.2 & \text{if } d \ge 500, \end{cases} \quad (20)$$

where α is the line slope and β is the ordinate at the origin.

The numerical values of this system are chosen so that the robot adopts the navigation agent when the distance from the target is less than 100 mm (U_n is maximal). When the distance varies between 100 and 50 mm, the utility function decreases linearly with d. When the distance becomes more than 500 mm, the value of U_n is minimal, which means that the navigation agent becomes disabled.

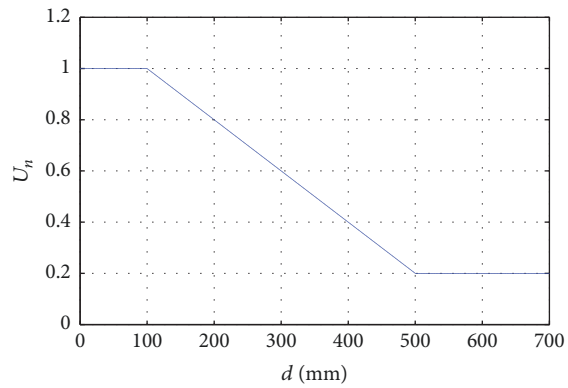

Figure 12: The utility function of the navigation agent.

The utility function related to the static and dynamic obstacle avoidance agents, respectively, U_{soa} and U_{doa} is combination between the distances which separate the robot from the static and dynamic obstacles, respectively, d_{so} and d_{do} and the orientation of the robot with respect to the static and dynamic obstacles, respectively, φ_{so} and φ_{do}. They are defined as

$$U_{\text{soa}} = U_{d_{\text{so}}} \times U_{\varphi_{\text{so}}},$$
$$U_{\text{doa}} = U_{d_{\text{do}}} \times U_{\varphi_{\text{do}}} \quad (21)$$

being

$$U_{d_{\text{so}}} = \begin{cases} 1 & \text{if } d_{\text{so}} \le R_c \\ \alpha_1 * d_{\text{so}} + \beta_1 & \text{if } R_c < d_{\text{so}} < 100 \\ 0.2 & \text{if } d_{\text{so}} \ge 100, \end{cases}$$
$$U_{d_{\text{do}}} = \begin{cases} 1 & \text{if } d_{\text{do}} \le 50 \\ \alpha_2 * d_{\text{do}} + \beta_2 & \text{if } 50 < d_{\text{do}} < 100 \\ 0.2 & \text{if } d_{\text{do}} \ge 100. \end{cases} \quad (22)$$

The same principle of the navigation agent is adopted here. For the dynamic obstacle, the robot enters to the dangerous zone when the distance from the obstacle is less than 50 mm. While for the static obstacle, the dangerous zone is defined as the convergence circle, which means that the robot should move far away when the distance from the obstacle is less than R_c, which is the radius of the convergence circle calculated previously (see Figures 13 and 14).

On the other hand, φ_{so} and φ_{do} are represented by the same function U_{φ_o}:

$$U_{\varphi_o} = \begin{cases} \alpha_3 * \varphi_o + \beta_3 & \text{if } -\frac{\pi}{2} \le \varphi_o \le -\frac{\pi}{6} \\ 1 & \text{if } -\frac{\pi}{6} \le \varphi_o \le \frac{\pi}{6} \\ -\alpha_4 * \varphi_o + \beta_4 & \text{if } \frac{\pi}{6} \le \varphi_o \le \frac{\pi}{2}. \end{cases} \quad (23)$$

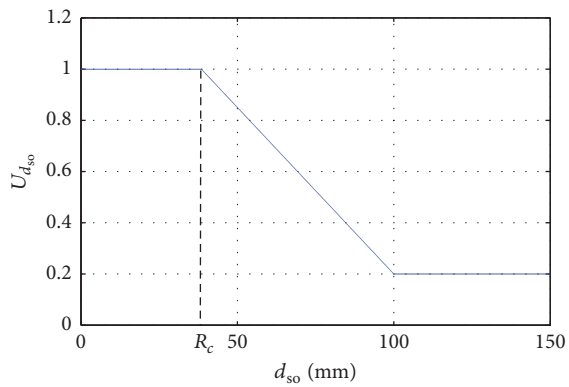

Figure 13: The utility function related to the distance of the static obstacle avoidance agent.

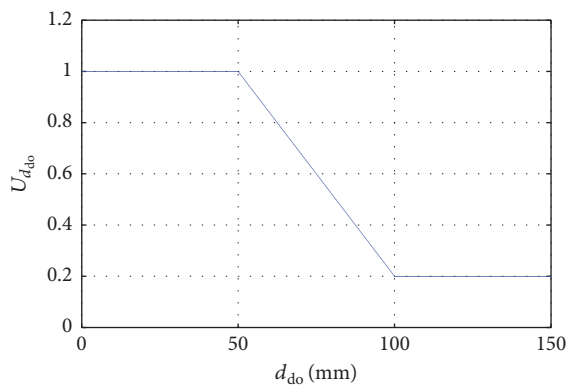

Figure 14: The utility function related to the distance of dynamic obstacle avoidance agent.

This utility function is illustrated in Figure 15.

α_i $(i = 1, \ldots, 4)$ is the line slope, and β_i is the ordinate at the origin.

Once the values of the utility functions are obtained, a negotiation is made to choose which agent the robot selects. Actually several ways of action selection exist [29]. Here, the actual agent sends its value to the agent in conflict. The one having the higher value is the one who controls the robot.

5. Simulation Results

This section is dedicated to test the efficiency of the proposed method. So different configurations were tested. We begin with the simple one. We consider an environment containing a static obstacle, and we must prove that the robot must go around the obstacle in the case of clockwise or counterclockwise motion direction. For that purpose, we have proposed two configurations (see Figures 16 and 17). In both configurations, the robot starts from the initial position $(X, Y) = (0, 0)$ with two different initial directions ($\alpha = \pi/2$ and $\alpha = 0$) and reaches the goal $(X_t, Y_t) = (300, 300)$. The robot is able to avoid the static obstacle applying the sliding mode approach to track the cycle limit trajectory regardless of the robot motion direction (clockwise or counterclockwise).

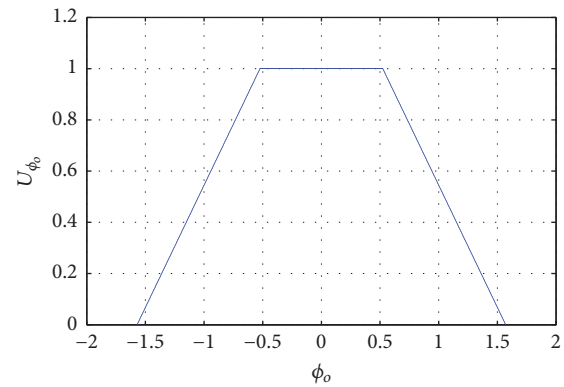

Figure 15: The utility function related to the angle of the dynamic obstacle avoidance agent.

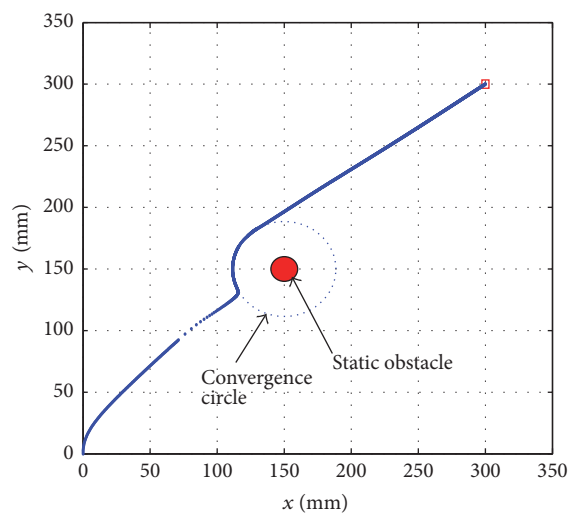

Figure 16: Simulation results with one static obstacle ($\alpha = \pi/2$).

Now, we treat the case of environments containing both static and dynamic obstacles.

In Figure 18, the coordination between the agents is verified in en environment containing a static obstacle and a dynamic obstacle which moves horizontally from an initial position $(X_{do}, Y_{do}) = (200, 100)$ to the left.

Figures 19 and 20 deal with the problem of cluttered environment. In Figure 19, the dynamic obstacle moves vertically from the point $(X_{do}, Y_{do}) = (75, 225)$ to the down. The configuration shows that the system can be adaptable to any change of the obstacle's position or number.

Figure 20 presents the case of highly complicated environments with several static and dynamic obstacles with different trajectories. The first dynamic obstacle starts from the position $(X_{do1}, Y_{do1}) = (300, 50)$, and the second obstacle moves from the point $(X_{do1}, Y_{do1}) = (50, 500)$. The robot detects any static obstacle that comes on its way and avoids it by tracking its convergence circle. For the dynamic obstacles, the robot is able to avoid them and reach the target, generating a smooth trajectory.

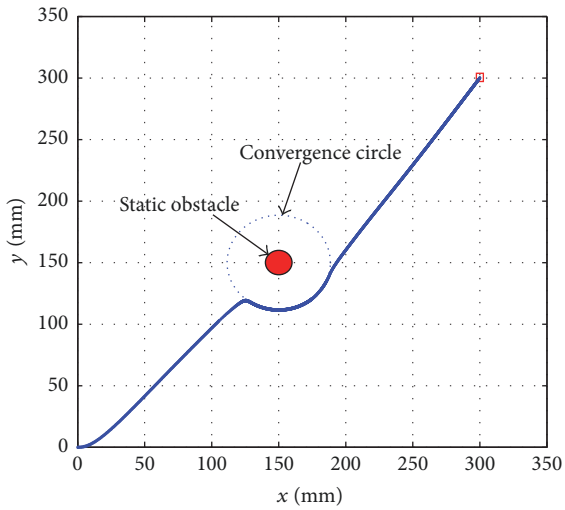

FIGURE 17: Simulation results with one static obstacle ($\alpha = 0$).

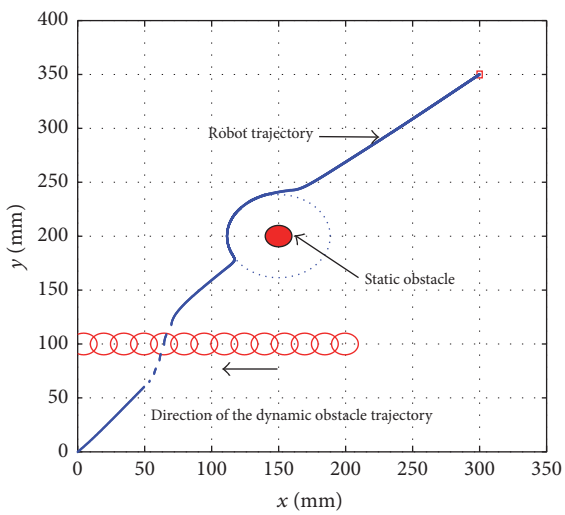

FIGURE 18: Simulation results for one dynamic obstacle and one static obstacle.

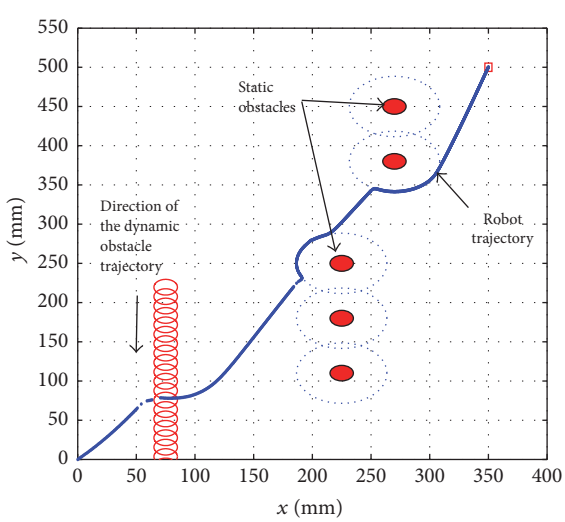

FIGURE 19: Simulation results in a cluttered environment.

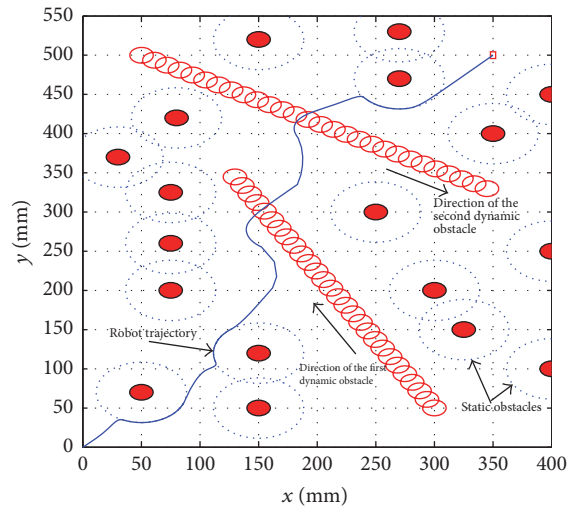

FIGURE 20: Simulation results in a cluttered environment.

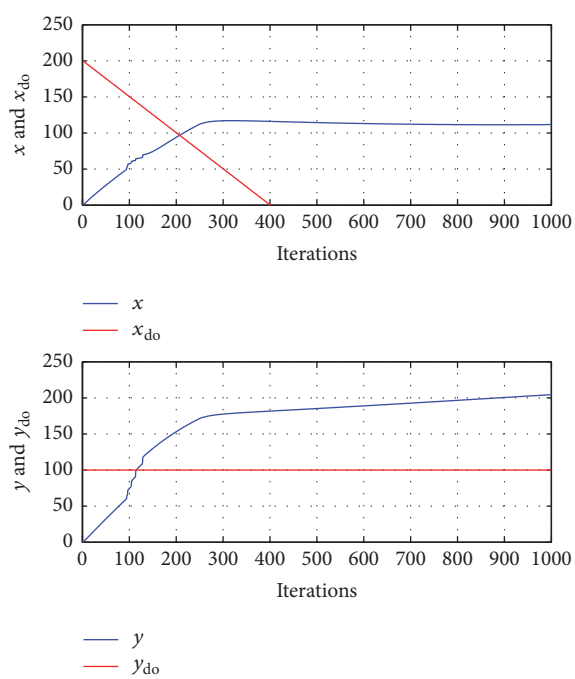

FIGURE 21: Coordinates of the robot and the dynamic obstacle according to time in the case of a horizontal mobile obstacle trajectory.

According to Figures 21, 22, and 23, there is no collision between the robot and the dynamic obstacle in the three previous simulations. The coordinates of the robot and the dynamic obstacles are not equal at the same time collision t, which means that the robot can move away perfectly from the dynamic obstacles.

The simulations presented in this section illustrate the efficiency of the proposed method and the ability of the robot to transit properly from an agent to another using the suggested utility functions and to reach the final destination safely.

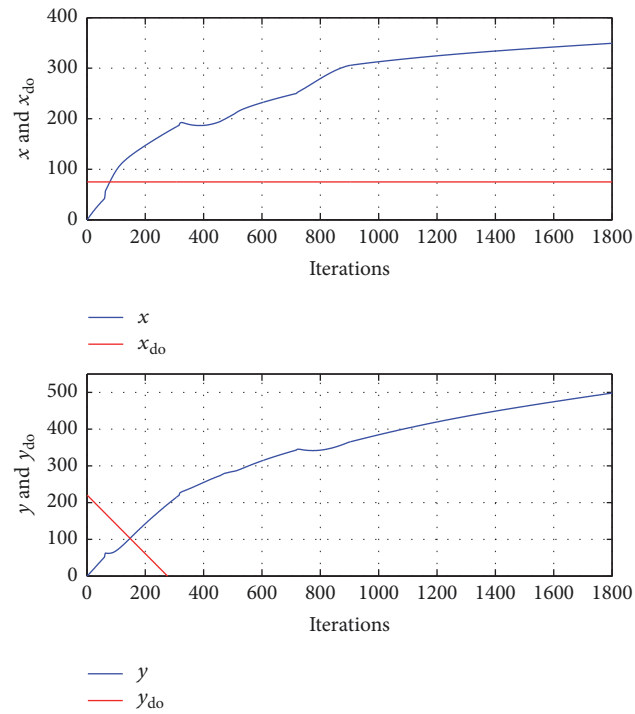

Figure 22: Coordinates of the robot and the dynamic obstacle according to time in the case of a vertical mobile obstacle trajectory.

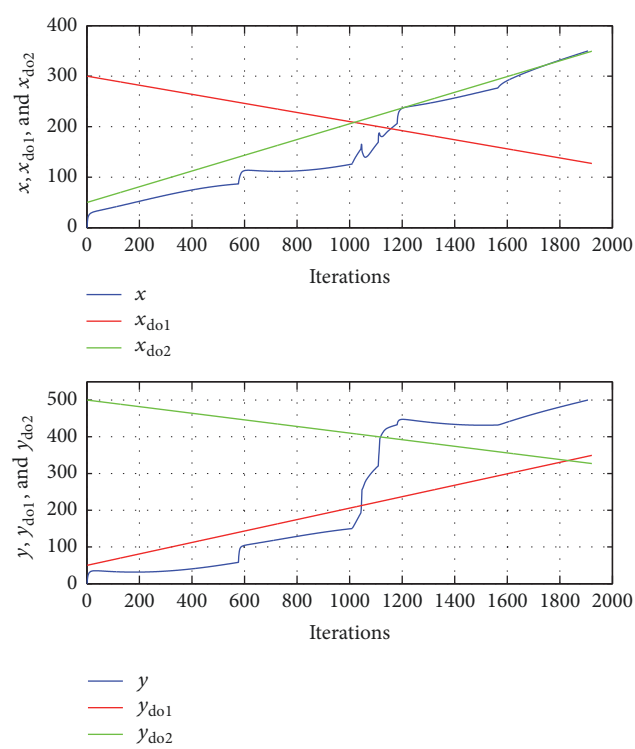

Figure 23: Coordinates of the robot and the dynamic obstacles according to time in the case of two different mobile obstacles trajectories.

6. Conclusion

On this work, a multiagent system is implemented for autonomous navigation of mobile robot in an environment containing static and dynamic obstacles. According to the data sent to the perception agent, a coordination system based on utility function is applied. Therefore, the suitable agent is chosen: navigation agent based on simple fuzzy system and dedicated to bring the robot to the target, static obstacle avoidance responsible for leading the robot away from static obstacles tracking an orbital trajectory with the sliding mode control, and dynamic obstacle avoidance which has a role in avoiding dynamic obstacles applying hierarchical fuzzy system. Some tests are made and presented to test the validity of the proposed method.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] Z. A. Baig, "Multi-agent systems for protecting critical infrastructures: a survey," *Journal of Network and Computer Applications*, vol. 35, no. 3, pp. 1151–1161, 2012.

[2] B. Innocenti, B. López, and J. Salvi, "A multi-agent architecture with cooperative fuzzy control for a mobile robot," *Robotics and Autonomous Systems*, vol. 55, no. 12, pp. 881–891, 2007.

[3] M. Boujelben, C. Rekik, and N. Derbel, "A multi-agent architecture with hierarchical fuzzy controller for a mobile robot," *International Journal of Robotics and Automation*, vol. 30, no. 3, pp. 289–298, 2015.

[4] C. G. Cena, P. F. Cardenas, R. S. Pazmino, L. Puglisi, and R. A. Santonja, "A cooperative multi-agent robotics system:Design and modelling," *Expert Systems with Applications*, vol. 40, no. 12, pp. 4737–4748, 2013.

[5] A. Hentout, B. Bouzouia, and Z. Toukal, "Multi-agent architecture model for driving mobile manipulator robots," *International Journal of Advanced Robotic Systems*, vol. 5, no. 3, pp. 257–268, 2008.

[6] M. Kolp, P. Giorgini, and J. Mylopoulos, "Multi-agent architectures as organizational structures," *Autonomous Agents and Multi-Agent Systems*, vol. 13, no. 1, pp. 3–25, 2006.

[7] Y. Ono, H. Uchiyama, and W. Potter, "A mobile robot for corridor navigation: A multi-agent approach," in *Proceedings of the 42nd Annual Southeast Regional Conference, ACM-SE 42*, pp. 379–384, USA, April 2004.

[8] D. Busquets, C. Sierra, and R. López de Màntaras, "A multiagent approach to qualitative landmark-based navigation," *Autonomous Robots*, vol. 15, no. 2, pp. 129–154, 2003.

[9] A. Benzerrouk, L. Adouane, and P. Martinet, "Stable navigation in formation for a multi-robot system based on a constrained virtual structure," *Robotics and Autonomous Systems*, vol. 62, no. 12, pp. 1806–1815, 2014.

[10] N. Hacene and B. Mendil, "Toward safety navigation in cluttered dynamic environment: A robot neural-based hybrid autonomous navigation and obstacle avoidance with moving target tracking," in *Proceedings of the 3rd International Conference on Control, Engineering and Information Technology, CEIT 2015*, dza, May 2015.

[11] R. Rojas and A. G. Förster, "Holonomic control of a robot with an omnidirectional drive," *KI-Künstliche Intelligenz*, vol. 20, no. 2, pp. 12–17, 2006.

[12] J. H. Lee, C. Lin, H. Lim, and J. M. Lee, "Sliding mode control for trajectory tracking of mobile robot in the RFID sensor space," *International Journal of Control, Automation, and Systems*, vol. 7, no. 3, pp. 429–435, 2009.

[13] I. Engedy and G. Horváth, "Artificial neural network based mobile robot navigation," in *Proceedings of the WISP 2009 - 6th IEEE International Symposium on Intelligent Signal Processing*, pp. 241–246, Hungary, August 2009.

[14] O. Khatib, "Real-time obstacle avoida nce for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[15] M. Boujelben, C. Rekik, and N. Derbel, "Hierarchical fuzzy controller to avoid mobile obstacle for a mobile robot," in *Proceedings of the 2013 10th International Multi-Conference on Systems, Signals and Devices, SSD 2013*, tun, March 2013.

[16] 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 Transactions on Fuzzy Systems*, vol. 19, no. 2, pp. 379–392, 2011.

[17] D. Ayedi, M. Boujelben, and C. Rekik, "nterval type-2 tsk fuzzy approach for autonomous mobile robot control in presence of uncertainties," in *Proceedings of the Systems, Signals & Devices (SSD), 2017 14th International Multi-Conference*, pp. 280–287, 2017.

[18] D. Nakhaeinia, S. H. Tang, S. B. Mohd 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.

[19] C.-W. Chen, "Interconnected TS fuzzy technique for nonlinear time-delay structural systems," *Nonlinear Dynamics*, vol. 76, no. 1, pp. 13–22, 2014.

[20] C. Rekik, M. Djemel, and N. Derbel, "A discussion on the optimal control of a robot manipulator by a hybrid genetic-fuzzy controller," *International Journal of Robotics and Automation*, vol. 23, no. 3, pp. 150–159, 2008.

[21] M. Boujelben, C. Rekik, and N. Derbel, "Hierarchical fuzzy controller for a nonholonomic mobile robot," in *Proceedings of the 2012 20th Mediterranean Conference on Control and Automation, MED 2012*, pp. 341–347, esp, July 2012.

[22] M. Boujelben, C. Rekik, and N. Derbel, "A hybrid fuzzy-sliding mode controller for a mobile robot," *International Journal of Modelling, Identification and Control*, vol. 25, no. 3, pp. 155–164, 2016.

[23] G. Antonelli and S. Chiaverini, "Experiments of fuzzy lane following for mobile robots," in *Proceedings of the 2004 American Control Conference (AAC)*, pp. 1079–1084, usa, July 2004.

[24] L. Adouane, "Orbital obstacle avoidance algorithm for reliable and on-line mobile robot navigation," in *Proceedings of the in 9th Conference on Autonomous Robot Systems and Competitions*, 2009.

[25] D.-H. Kim and J.-H. Kim, "A real-time limit-cycle navigation method for fast mobile robots and its application to robot soccer," *Robotics and Autonomous Systems*, vol. 42, no. 1, pp. 17–30, 2003.

[26] R. Solea, A. Filipescu, and U. Nunes, "Sliding-mode control for trajectory-tracking of a wheeled mobile robot in presence of uncertainties," in *Proceedings of the 7th Asian Control Conference (ASCC '09)*, pp. 1701–1706, Hong Kong, August 2009.

[27] I. González, S. Salazar, and R. Lozano, "Chattering-free sliding mode altitude control for a quad-rotor aircraft: Real-time application," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1-4, pp. 137–155, 2014.

[28] M. Boujelben, C. Rekik, and N. Derbel, "Hierarchical neural network control for a mobile robot," in *Proceedings of the 2nd International Conference on Automation,Control, Engineering and Computer Science (ASECS)*, 2015.

[29] P. Pirjanian, "Behavior coordination mechanisms-state-of-the-art," Technical Report IRIS-99-375, University of Southern California, Institute for Robotics and Intelligent Systems, 1999.