Navigation of a two-wheel differential drive robot in a partially unknown environment

Breno P. de Meneses * **Gabriel H. V. da Silva** * **Lara R. Sobral** *
Mateus S. Marques * **Rodrigo T. de Araujo** * **Antonio M.N. Lima** **

* *DEE-UFCG*
** *Department of Electrical Engineering (DEE), Universidade Federal de Campina Grande (UFCG), PB, Brazil*
(breno.meneses@de.ufcg.edu.br)

Abstract: This paper proposes the use of a behavior-based control strategy for collision-free navigation of a two-wheel differential drive mobile robot in partially unknown environments. The proposed strategy is hierarchical, and the design of the high-level control layer (behavioral space) is based on Mamdani fuzzy logic inference, while the design of the low-level control layer (joint space) is based on Lyapunov stability theory. The collision and deadlock avoidance technique is based on subgoals that are evaluated in the behavioral space and supplied as references to the joint space controller. The *in silico* test study was executed by using the CoppeliaSim which is a simulation framework used for the prototyping, development and verification of robot systems and algorithms that is widely used by the robotics community. Several (robot initial and final poses, walls, corridors, make-like regions, and loose objects placement) test scenarios, and a comparative study with previous and related work were performed. The test results show the proposed strategy provided a smooth and shorter path in all cases. This improvement is basically related with the combined use of the subgoal technique and a Lyapunov based controller. In summary, the results corroborate the correctness of the methodology adopted in the design of collision-free navigation for differential drive mobile robots.

Keywords: Behavior-Based Systems, Sensor-based Control, Control Architectures, Programming, Fuzzy Control and Nonlinear Systems

1. INTRODUCTION

An autonomous mobile robot is a programmable device that uses proprioceptive and exteriorceptive sensor measurements to gather data from the surrounding environment, and process it to plan its motion and execute given tasks, and at the same time avoid collisions with objects that share the same space. Whenever a detailed map of the environment is known before hand, one may plan the robot's navigation to avoid collisions with static objects. Besides, the map-based navigation, rely on an accurate and up-to-date representation of the environment, which may not be available in unknown or partially unknown environments.

If unpredictable or dynamic objects appear in the robot's path, the navigation planner must include a collision avoidance strategy based on the on-board robot sensing capabilities. This is true in applications like, for instance, autonomous vehicles, warehouse automation, robot vacuum cleaners, to name just a few. When a mobile robot navigates autonomously in any given area, at least a collision and deadlock avoidance technique is required to ensure safe and efficient functioning.

In general, when an autonomous robot must navigate in a relatively complex and unknown environment, one prefers navigation strategies based on reactive control concepts (De Silva and Ekanayake, 2008). The combination of the different behaviors (behavior-based control) that the robot should exhibit in a given environment is a relatively simple and intuitive reactive control strategy. In the behavior-based control, one chooses a set of robot behaviors, e.g., "goal seeking", "obstacle avoidance", "deadlock disarming", and a technique to combine them. The selection of a given behavior is evaluated based on the measurements provided by the onboard robot sensors. Fuzzy Logic (Bao et al., 2009; Van Nguyen et al., 2017a,b), Arbitration (Brooks, 1986; Baumann et al., 2022), Blending (Adriansyah, 2014; Ramakrishna Pandian et al., 2021), Potential Fields (Khatib, 1986; Kim et al., 2016), Behavior Trees (Colledanchise and Ögren, 2018), Finite State Machines (Petrovic, 2008; Bozzi et al., 2022), Reinforcement Learning (Cherroun and Boumehraz, 2012; Sutton and Barto, 2018) and Evolutionary Algorithms (Shen, 2013; Sathiya and Chinnadurai, 2019) have been used individually or in combination to design behavior-based control systems for mobile robots, allowing for flexible and adaptive robot behaviors in various environments and tasks.

Among the mentioned methods, fuzzy logic was chosen because it provides an effective technique to deal with uncertainty by means of flexible linguistic categories and

⋆ This work was carried out with the support of the Department of Electrical Engineering, Center for Electrical Engineering and Informatics (CEEI), Federal University of Campina Grande, Campina Grande, PB.

logical rules of inference. This reduces the difference between human reasoning and numerical application, making it possible to define control systems through linguistic rules, which are adapted according to behavior, as in this context. In addition to its effectiveness as a control technique, fuzzy logic requires less formal knowledge of the plant and the environment in which it will be applied, in comparison to the so-called standard control system approach, allowing specific movements to be carried out with a simplified implementation.

Fuzzy behavior-based architectures for two-wheel differential drive robot navigation in partially unknown environments with static objects are proposed in (Bao et al., 2009) and (Van Nguyen et al., 2017a,b). The so called subgoal technique has been proposed in (Ye and Webb, 2009) for performance optimization. In general, the formulation of a behavior-based control problem does not deal with the low-level control, i.e., robot's joints control. Without the robot's low-level control layer there is no guarantee that the actual motion of the robot will be exactly as smooth as expected by the behavioral control layer (Panahandeh et al., 2019). The use of Lyapunov-based low-level controller for moving target tracking with a fuzzy strategy for obstacle avoidance has been proposed in (Benbouabdallah and Zhu, 2013; Kubo et al., 2020). However, the collision avoidance strategy is restricted to point objects, and thus cannot deal with deadlocks.

In this paper, we include a high-level control layer based on Fuzzy Logic that exploits the subgoal approach to determine robot motion routes. Also in this layer, the fuzzy controller allows the vehicle to develop different types of behaviors (e.g., obstacle avoidance, tracking, and deadlock disarming) while heading to a goal (go to goal). The design of the low-level control layer is based on Lyapunov stability theory and provides robust stability and convergence.

2. PROBLEM DESCRIPTION

Figure 1 shows a schematic illustration of a robot with two-wheel differential drive, a final goal, and an environment with an obstacle, supposedly static and unknown. The objective of this paper is to design a control strategy that allows the robot to go to the final goal while safely avoiding collision in its path. To do this, the following assumptions are made

- The goal (target) and robot pose is known and specified, meaning that we consider a partially unknown environment.
- The obstacles in the environment are unknown to the robot, meaning it does not have prior information about their locations or shapes.
- The robot operates in a two-dimensional space, as indicated by the schematic illustration in Figure 1.
- The robot is equipped with range sensors that provide enough information for detecting and localizing obstacles.
- The robot is a non-holonomic system, and no lateral slip motion is allowed. However, it can move forward, backward, and rotate clockwise and counterclockwise around the z-axis (perpendicular to the xy plane depicted in Figure 1).

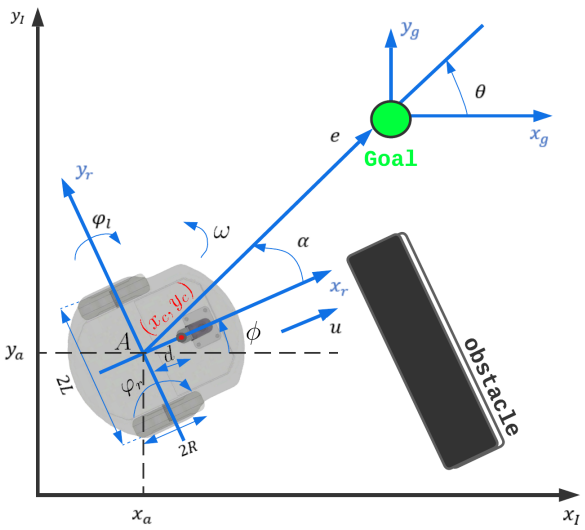

Figure 1. Schematic of the position and orientation of the vehicle with respect to the final goal.

Besides the basic assumptions regarding the environment, we also consider that

- For designing both the high level and the low-level controller, one considers that the two-wheel differential drive robot is represented by the so called unicycle model, which is a third-order two-input non-linear kinematic model (Lynch and Park, 2017); from now on we will denote this model as our action model.
- For testing the design solution one considers that the two-wheel differential drive robot with casters (off-centered orientable, not driven wheel joint) is represented by a dynamic model (mass, moment of inertia, friction), which is a fourth-order two-input non-linear model (Dhaouadi and Hatab, 2013); from now on we will denote this model as our knowledge model.
- The testing environment (a virtual world scene including the floor, walls, corridors, maze-like regions, obstacles, range sensors, and robot) and the knowledge model are simulated by using the latest version of CoppeliaSim (Rohmer et al., 2013). CoppeliaSim is a simulation framework used for the prototyping, development and verification of robot systems and algorithms that is widely used by the robotics community.

The proposed control strategy is hierarchical. In the upper layer, the behavior-based controller combines the behaviors and sends them to the lower layer, where a nonlinear controller, tuned by using a genetic algorithm, is used.

3. LOW-LEVEL CONTROL LAYER

3.1 Lyapunov Controller

Consider a vehicle positioned at a nonzero distance with respect to a goal pose, as depicted in Figure 1. As Kubo et al. (2020) and Ravangard (2015) a Lyapunov controller can be used to describe a unicycle-like motion given by

$$\begin{cases} \dot{x} &= u\cos(\phi) \\ \dot{y} &= u\sin(\phi) \\ \dot{\phi} &= \omega \end{cases} \quad (1)$$

where x, y, denote the vehicle position, and ϕ its orientation. The control inputs are the linear velocity, u, and the rotational velocity ω, with respect to the $x_I \times y_I$ frame. However, for designing the non-linear control law the vehicle motion will be represented by

$$\begin{cases} \dot{e} = -u\cos(\alpha) \\ \dot{\alpha} = -\omega + u\frac{\sin(\alpha)}{e} \\ \dot{\theta} = u\frac{\sin(\alpha)}{e} \end{cases} \quad (2)$$

where e and α denote the distance and orientation with respect to the goal, respectively. Now by choosing following Lyapunov candidate function

$$V(e,\alpha) = \frac{1}{2}\lambda e^2 + \frac{1}{2}h\alpha^2, \lambda > 0, h > 0 \quad (3)$$

Feedback control laws can be found by minimizing their time derivatives, causing them to asymptotically converge to zero and become independent of the values of λ and h.

$$\begin{cases} u = \gamma e\cos(\alpha) \\ \omega = k\alpha + \gamma\cos(\alpha)\sin(\alpha) \end{cases} \quad (4)$$

These equations guarantees global stability and the boundedness of the state trajectory corresponding to any bounded initial condition for $\kappa > 0$ and $\gamma > 0$ (Aicardi et al., 1995).

It is worth to point out that the error variables e and α used to compute u and ω depend on the actual goal being tracked. Besides, one has to use the following equations

$$\begin{cases} \omega_r = \frac{2u+\omega L}{2R} \\ \omega_l = \frac{2u-\omega L}{2R} \end{cases} \quad (5)$$

to determine the left (ω_r) and right (ω_r) wheel joint velocities (L and R are defined in Figure 1).

The use of a nonlinear low-level controller has been proposed by Benbouabdallah and Zhu (2013). However, their control law is given by

$$\begin{cases} u = v_T\frac{\cos(\beta)}{\cos(\alpha)} - K_v e_D\cos(\alpha) \\ \omega = -K_w\alpha - \frac{v}{D}\sin(\alpha) + \frac{v_T}{D}\sin(\beta) \end{cases} \quad (6)$$

where

$$\begin{cases} \alpha = \theta - \Phi \\ \beta = \theta_T - \Phi \\ e_D = D_d - D \end{cases} \quad (7)$$

Please see (Benbouabdallah and Zhu, 2013) for the meaning of the terms of the control law. To justify our choice of the control law given in (4) instead of the one given in (6), we conducted a preliminary comparison with a simple scene containing only loose points-objects. Figure 2 shows this simple scene and the robot navigation under both the [S_{new}-(4)] and [S_{old}-(6)] control laws. In Figure 2 the red path is associated with S_{new} and the blue path is associated with S_{old}. Table 1 show that the path length under S_{old} is larger than (≈1.9X) the one observed under S_{new}, while the path travel times are essentially the same. For the [S_{old}-(6)] we used $\theta_T = \theta = 90°$, $\alpha = \beta$, $K_v = 2.07$, $K_w = 1.49$, $D_d = 0.2$ m , and v_T= 0.15 m/s. Whereas, for the [S_{new}-(4)] we used $\gamma = 0.3$ and $\kappa = 1$.

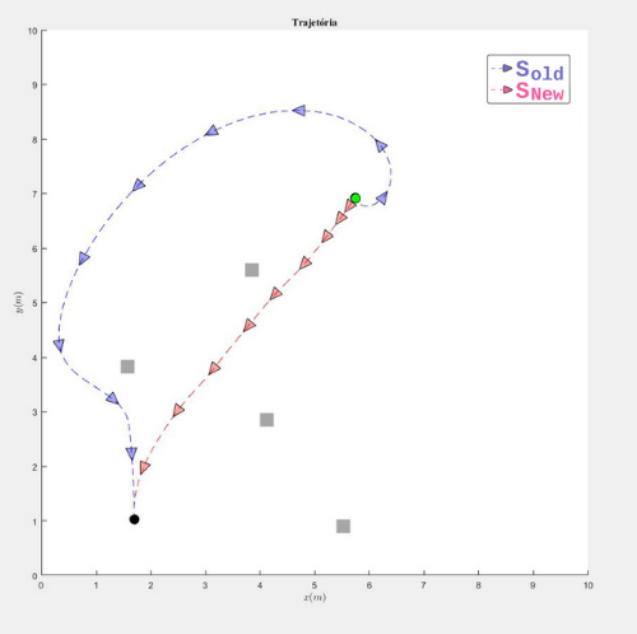

Figure 2. Simple scene containing only four loose points-objects. The red path is associated with S_{new} and the blue path is associated with S_{old}

Table 1. Path length and path travel time comparison (Figure 2)

	Path length (m)	Path travel time (s)
S_{old}	13.76	14.39
S_{new}	7.28	14.59

3.2 Controller gains tuning

The selection of the low-level controller gains was formulated as an optimization problem, and its solution was determined by using a genetic algorithm (Burjorjee, 2007). The two gains were coded into a 40-bits word, being 20-bits for κ and 20-bits for γ. The chosen test scenario is depicted in Fig. 3, and the fitness function was defined as

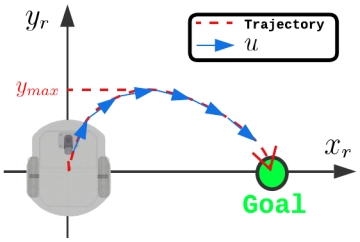

Figure 3. Test scenario used in the controller gains tuning procedure.

$$fitness = \frac{1}{\Phi_y} + \frac{1}{\Phi_u}, \quad (8)$$

where

$$\Phi_y = |\max(y) - y_{\text{ref}}| \text{ and } \Phi_u = |\max(u) - u_{\text{ref}}|. \quad (9)$$

This choice of this fitness function aims at ensuring that, during the robot's movement toward the target pose, $u < u_{\text{ref}}$ and $y < y_{\text{ref}}$ is always satisfied. In the test scenario, the initial robot pose is $(0, 0, \pi/2)$, and the target pose is $(0, 1, -\pi/2)$. The genetic algorithm was configured to

use the following hyperparameters: population size = 200, number of generations = 40, crossover probability = 1, and the mutation probability per bit = 0.003. The final gains were $\gamma = 0.328$ and $\kappa = 1.08$. For these gain values, the time-to-target is 5 s, $y_{\max} = 0.15$ m, and $u_{\max} = 1.2$ m/s.

4. HIGH-LEVEL CONTROL LAYER

The proposed fuzzy behavior-based architecture for mobile robot navigation is depicted in Figure 4. For implementing the high-level control layer strategy one assumes that the on-board robot sensing capability is a belt of 16 ultrasound range sensors disposed as depicted in Figure 5. However, the same concepts used here can be applied to other types of range sensors like for instance infrared and LIDAR.

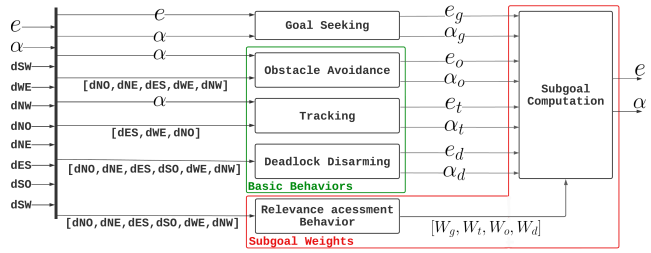

Figure 4. Fuzzy behavior-based architecture for mobile robot navigation.

4.1 Sensor arrangement

The sensors of the belt are grouped in eight groups, according the wind rose orientations, i.e., west, northwest, north, northeast, east, southeast, south and southwest, as shown in the Figure 5.

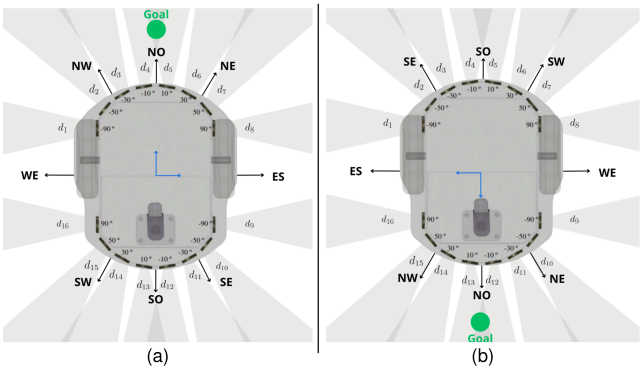

Figure 5. Placement of the ultrasound sensors on the robot chassis and assignment of the clusters based on the wind rose orientations. (a) Standard orientation; (b) Reorientation of the clusters to redefine the front of the robot based on the target orientation.

The distances from the robot to an obstacle (d_i) along the wind rose orientations are determined by combining the readings issued from two neighboring ultrasound sensors. At the time of its initialization, based on the quadrant of the goal, shown in the grouping sensor readings algorithm (Algorithm 1), the front of the robot is set, which will imply either positive (1st or 2nd quadrant) or negative (3rd or 4th quadrant) linear velocities. Thus, by this grouping sensor readings the north is always in the same direction of the motion, as shown in Figure 5.

Algorithm 1 Grouping sensor readings

Input : α_{goal}, $[d_1, \cdots, d_{16}]$
Output: Grouped distances along the wind rose with north in the direction of movement.
Require t_0

if $|\alpha_g| > \frac{\pi}{2}$ **then**
  dSW $\leftarrow \min(d_{14}, d_{15})$
  dWE $\leftarrow \min(d_1, d_{16})$
  dNW $\leftarrow \min(d_2, d_3)$
  dNO $\leftarrow \min(d_4, d_5)$
  dNE $\leftarrow \min(d_6, d_7)$
  dES $\leftarrow \min(d_8, d_9)$
  dSE $\leftarrow \min(d_{10}, d_{11})$
  dSO $\leftarrow \min(d_{12}, d_{13})$
  $u \leftarrow u$
else
  dSW $\leftarrow \min(d_6, d_7)$
  dWE $\leftarrow \min(d_8, d_9)$
  dNW $\leftarrow \min(d_{10}, d_{11})$
  dNO $\leftarrow \min(d_{12}, d_{13})$
  dNE $\leftarrow \min(d_{14}, d_{15})$
  dES $\leftarrow \min(d_1, d_{16})$
  dSE $\leftarrow \min(d_2, d_3)$
  dSO $\leftarrow \min(d_4, d_5)$
  $u \leftarrow -u$
return dSW, dWE, dNW, dNO, dNE, dES, dSE, dSO

4.2 Basic behaviors

The behavior determination is based on readings of the distance sensor groups, differing only in the quantity, number of groups and position of the sensors (Van Nguyen et al., 2017a,b; Kuo et al., 2013). In the proposed fuzzy architecture, the basic behaviors are "goal seeking", "obstacle avoidance", "tracking" and "deadlock". The membership functions used are summarized as shown in Figure 6. The control signals are the linear (u) and angular (ω) velocities. By using the low-level Lyapunov control law, it is possible to change the input variables for distance (e) and target orientation (α), as if the goal was elsewhere.

The linguistic labels for the distances issued from the ultrasound range sensor and to the target are near (N), medium (M) and far (F), as shown in Figure 6 (a), (c) and (d). The target distance range was calculated so that it would not saturate linear velocity of the robot. For orientation, the inspiration of wind roses is also used, where SOp and SOn are the two halves representing the SO, observable in Figure 6 (b).

The rules that define each behavior of the proposed architecture can be summarized as follows:

Goal-Seeking: It is not necessary to implement the fuzzy behavior to achieve the goal in the absence of obstacles. This is because its input, the distance and orientation to the goal, can be imposed on the low-level controller without the need to perform inference. In this way, the subgoal pointed by this behavior is oriented in the same direction as the main goal and located within a defined radius;

Obstacle Avoidance: The fuzzy behavior designed for obstacle collision avoidance does not use the distance read-

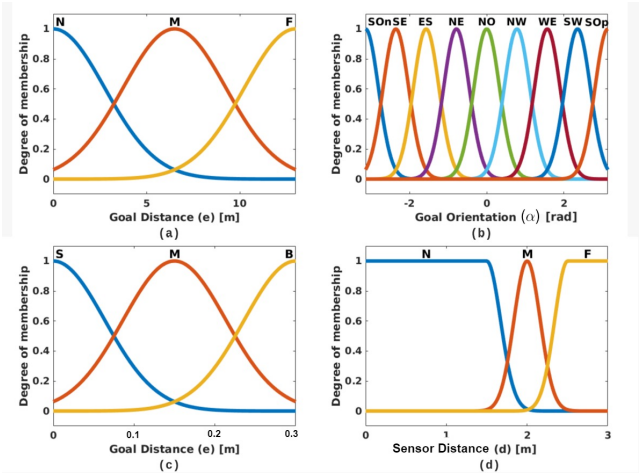

Figure 6. The linguistic labels and Gaussian membership functions (a) Goal distance; (b) Goal orientation; (c) Goal distance output of Deadlock behavior; (d) Wind rose distances from WE, NW, NO, NE, ES, SE, SO and SW sensor clusters.

ings of all clusters, as this would increase the number of rules, only those arranged in the direction of movement of the robot (WE, NW, NO, NE and ES). Also, to obtain a possible shortest trajectory, the goal direction is always considered. It is important to note that the front distance (NO) needs to be small, so small that this behavior is activated, and its rules are derived from this initial condition. Thus, the subgoals defined by this behavior are always oriented either WE or ES of the robot, depending on the goal orientation and position of the obstacle, i.e. , during goal-seeking behavior an L-shaped front and left (right) side obstacle is identified, the subgoal is positioned a short distance in the direction of ES (WE);

Tracking: An important behavior to design is tracking, which generally consists of unilateral and bilateral wall-following behaviors, i.e. corridors. This behavior should be complementary to obstacle avoidance behavior, avoiding strictly lateral collisions and following them. Thus, right-angle lateral sensor groups (WE and ES) are evaluated, provided there is no frontal obstacle (NO), and target orientation so that it does not follow corridors when unnecessary. Thus, the subgoals will always be within a small distance in the NO, WE or ES direction, when the sub-object is behind the wall or in front of it, to the left or right, respectively;

Deadlock: Deadlock disarming is important for solving dead-end corridors and u-shaped obstacles. It is possible to find this solution without describing a specific behavior, merely avoiding obstacles and following the trail. However, for better performance, we specify this behavior. In this paper, a rotation on its own axis clockwise or counterclockwise is proposed. In the case where the front and side sensor groups (NO, WE, and ES) are triggered simultaneously and the distance to the obstacle is small, a subgoal is defined such that the motion is towards SO, and this eventually leads the robot to make a rotation clockwise or counterclockwise.

4.3 Subgoals generator

The subgoal generator yields a position relative to the robot's frame as input for the low-level velocity controller, rather than directly producing linear and angular velocities as outputs from the fuzzy controller. The inputs are the polar coordinates error variables $[e, \alpha]$, as explained in section 3. The subgoal is determined based on the rules defined for each behavior in the modular fuzzy controller. The rules that define the subgoal for each of the considered behaviors were summarized in the Section 4.2.

By means of the subgoal generator, new positions will be generated as virtual goals for the robot to act according to that particular condition. This can be seen as the robot seeking multiple sequential targets that collectively will form the path leading to the final target, as depicted in Figure 7.

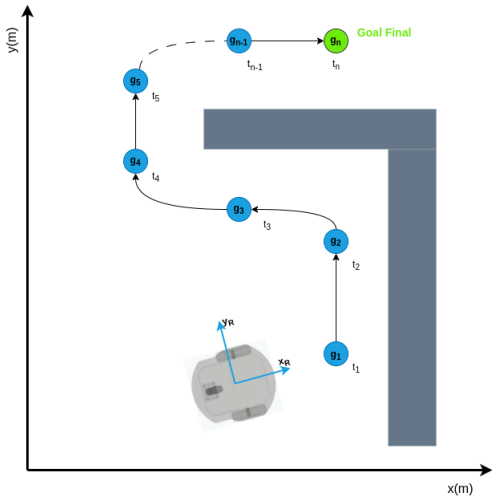

Figure 7. Virtual goals issued by the subgoal generator $g_1 = (t_1, e_1, \alpha_1), \cdots, g_{n-1} = (t_{n-1}, e_{n-1}, \alpha_{n-1})$ towards the final target but avoiding collisions.

For combining the behaviors we also used fuzzy logic. The input for this block are the wind rose distances from the robot to an obstacle while the outputs are the weights that each behavior will have in determining the subgoal that will be provided to the low-level controller. Although the fuzzy rules be based on (Bao et al., 2009), in the present solution there is no discontinuity since the behavior transition is gradual and thus avoids speed spikes in the robot motion. This smoothness can be inferred from the graphical representation of the rule surface depicted in Figure 8.

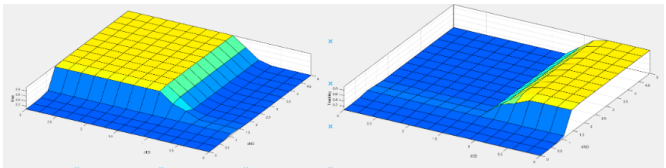

Figure 8. Rule surface for the fuzzy logic controller used to combine behaviors.

With $\{e_i, \alpha_i, W_i\}$, $i \in \{g, r, o, d\}$ the weighted sums are computed by

$$e(t_k) = \sum_i W_i(t_{k-1})\, e_i(t_{k-1})$$
$$\alpha(t_k) = \sum_i W_i(t_{k-1})\, \alpha_i(t_{k-1}) \quad (10)$$

at each sampling instant (t_k, $k = 1, 2, \cdots$) to generate the new subgoal along the final goal. The subscripts g, r, o and d denote the "Goal-Seeking", "Tracking", "Obstacle Avoidance" and "Deadlock Disarming" behaviors, respectively. Each blue circle in the Figure 9 image represents the virtual goal generated by using (10).

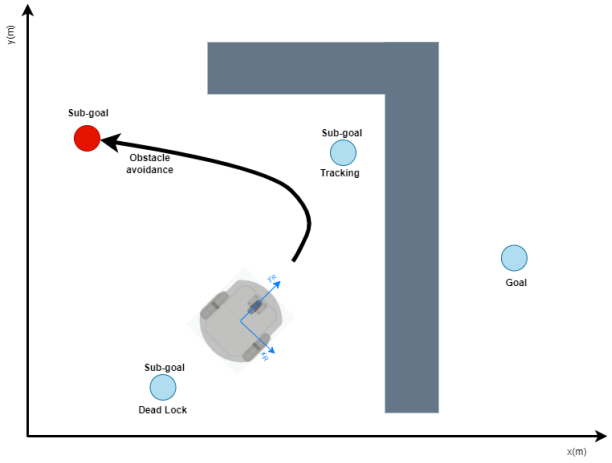

Figure 9. Representation of the position of the virtual goals generated by each of the behaviors

5. *IN SILICO* TESTS

To show the effectiveness of the control strategy proposed in this work, a model of the mobile robot, the surrounding 3D environment and the obstacles were instantiated on a virtual prototyping platform called CoppeliaSim Edu V4.5.1 (Rohmer et al., 2013). This open-source platform is widely used by the robotics community and allows the creation of scripts for remote communication with the created scene, controlling individually the elements of the scene and allowing the calculation of the robot movements, simulating the operation of the on-board range sensors and controlling interactions of the robot with obstacles, floors and other relevant elements. The physics engine used was the Bullet Physics Library (Coumans, 2015), is a robust and versatile physics simulation library that offers high-performance collision detection, rigid body dynamics, and constraint solving capabilities. Finally, we used the Pioneer 3DX (P3DX) from Adept Mobile Robots, which is a two-motor, two-wheeled differential drive robot with caster wheel (Robotics, 2006).

For the *in silico* tests it was necessary to create virtual worlds (scenes) where the P3DX is inside a room that contains several obstacles placed at different positions as shown in Figs. 10(first scene)-(sixth scene). For simplicity we considered that the instantaneous P3DX pose as obtained from the ground truth provided by CoppeliaSim is sent to the control strategy, but on the hand the information regarding the obstacles is hid.

The parameters of the P3DX are $R = 0.0975$ m, $L = 0.3810$ m, $u_{\max} = 1.2$ m/s, and $\omega_{\max} = 4.3$ rad/s (Robotics, 2006). The ultrasonic sensors are arranged around the robot chassis to provide a 360° viewing angle, as shown in Figure 5. All sensors in the belt are considered to have the same sensing characteristics, i.e., a detection range of 3 m and a field of view cone of 15°. The gains of the low-level controller were chosen to be $\gamma = 0.3$ and $\kappa = 1$, and were kept constant in all test conditions.

For the sake of comparison, we implemented the work of Bao et al. (2009) (S_{old}) and compared it with the solution proposed here (S_{new}) under the same scenarios to show the differences with respect to the smoothness and length of the robot's path. The three test scenarios are essentially the same used in Bao et al. (2009) and Van Nguyen et al. (2017a,b), but we changed the initial position and orientation of the robot, as well as of the goal. The reason for choosing these scenarios for demonstration is due to their different complexities and similarity with actual scenarios, to cover the different behaviors described previously.

Figures 10 (a) and (d) are relatively simple environments without loose obstacles other than the walls. It can be observed that in both cases S_{old} and S_{new} the robot successfully navigates, avoiding the obstacles and reaches the goal. However, when comparing the robot paths (Horiuchi and Noborio, 2001), it is possible to identify that under S_{new} the path length is shorter and smoother than the one observed under S_{old}. Table 2 Scene a show that the path length and path travel time under S_{old} are larger than (≈2X) as the ones observed under S_{new}

Table 2. Path length and path travel time comparison (Figure 10)

		Path length (m)	Path travel time (s)
Scene a	S_{old}	21.58	43.20
	S_{new}	10.81	22.24
Scene b	S_{old}	16.26	44.55
	S_{new}	13.97	37.45
Scene c	S_{old}	14.27	27.70
	S_{new}	13.34	33.29
Scene d	S_{old}	15.90	41.80
	S_{new}	13.89	38.395

Although having the same starting point (Figures 10 (a)), the robot's orientation under S_{new} is rotated by 180°, highlighting the efficacy of using the evaluation of the quadrant of the objective in the definition of the robot's front, performing the proper behavior configurations, and preventing local minimum.

Figures 10 (b) and (d) show a maze-like situation, .i.e, a nearly closed environment with restricted space for making turns. This type of scenario is built in order to emphasize the smoothness of the robot's movement. When comparing the trajectories under S_{old} and under S_{new}, we see a remarkable difference. In the former, the curves are sharper, almost scraping the walls, and susceptible to entry into a loop since the walls are very close, resulting in a sequence of behaviors that do not reach the solution. On the other hand, under S_{new} the path is smooth, with movements more compatible with obstacle placement. For the same scenario, a new position for the final goal was assigned so that S_{new} decides to navigate backwardly. In these cases, we can see less effort to align the robot's pose

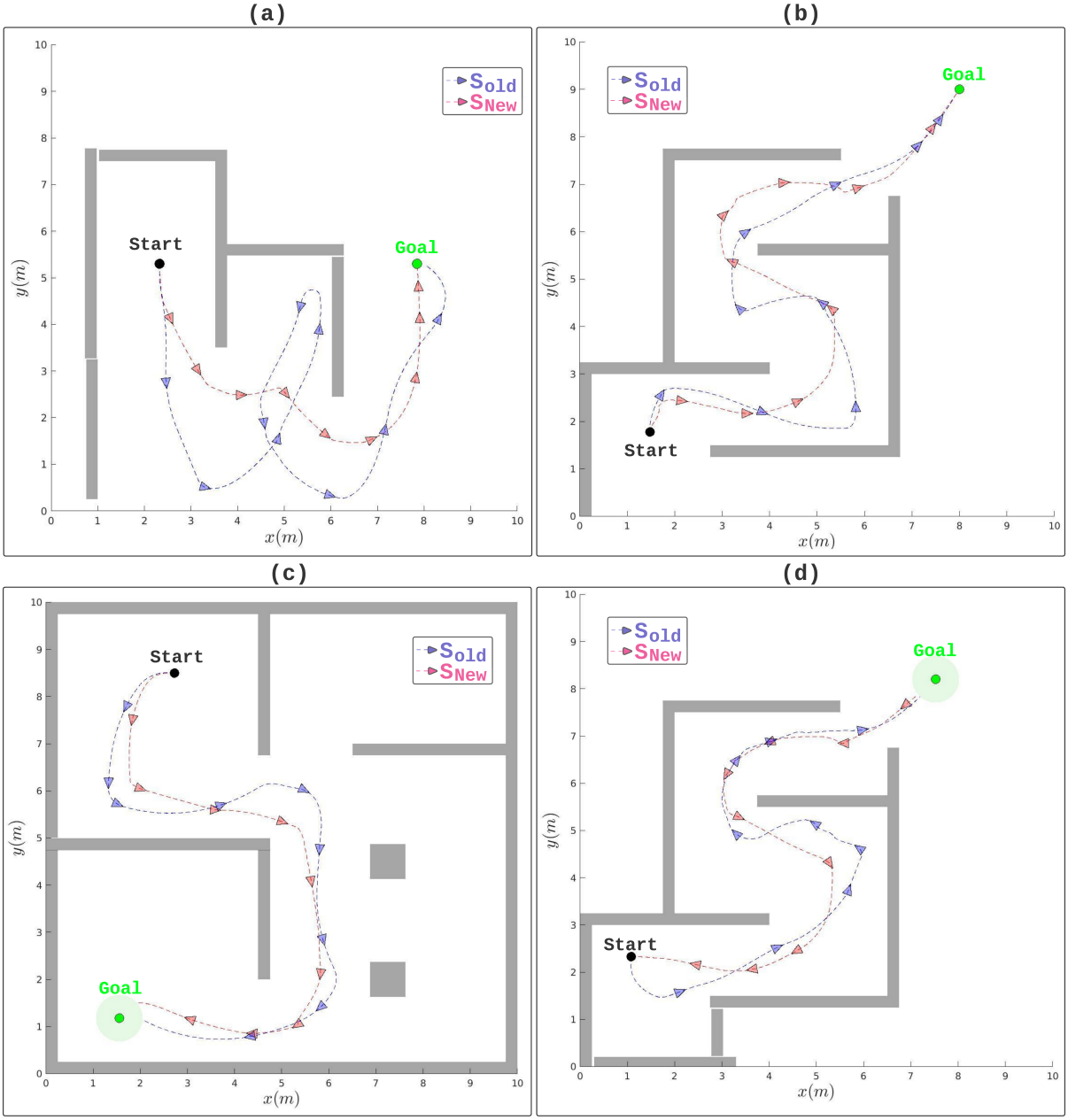

Figure 10. Comparison with the robot motion (colored triangles) without (S_{old}) and with the low-level control layer (S_{new}) in different scenarios, start point and end goal. The low-level control design is based on Lyapunov stability theory. (a) Corridor scenario; (b) Maze scenario with two exits; (c) Room scenario with goal in another room; (d) Maze scenario with one exit.

with the goal pose, and the robot navigates backwardly. Table 2, Scene b show that the path length and path travel time under S_{old} are larger than (>18%) the ones observed under S_{new}

Figures 10 (c), show environments close to real situations that rely on room sets and common areas with smaller objects compared to walls, such as chairs and centers. In these cases, under both S_{old} and S_{new} to robot reach the objective. Again, under S_{new} the motion is smoother with fewer abrupt curves and far from the walls. As in the other cases, the goal was allocated in such a position that the robot under S_{new} navigates backwardly, and furthermore, it was placed between two less problematic objects in terms of convergence. [1]

6. CONCLUSION

This paper has presented a behavior-based control strategy for collision-free navigation of a two-wheel differential drive mobile robot in partially unknown environments. The proposed strategy is hierarchical, and the design of the behavioral layer controller is based on fuzzy logic inference, while the design of the wheel joint controller is based on Lyapunov stability theory. We have also compared

[1] Video demonstration: Robot navigating under S_{new} in Coppeliasim.

two different nonlinear control laws for the wheel joint controller. At the behavioral level, the collision and deadlock avoidance is achieved by means of selecting proper subgoals towards the final. The *in silico* study revealed that the proposed strategy yields a mobile robot motion that is smooth, short, and collision and deadlock free. Some future works include the validation and refinement of the proposed solution by using a laboratory prototype, extending the study to consider the dynamic model of the obstacle and mobile robot for dealing with energy constraints, and modifying the low-level control layer gains according to prescribed subgoals.

REFERENCES

Adriansyah, A. (2014). Design of context dependent blending (cdb) in behaviour based robot using particle swarm fuzzy controller (psfc). In *1st International Conference on Computer Science and Engineering*. Sriwijaya University.

Aicardi, M., Casalino, G., Bicchi, A., and Balestrino, A. (1995). Closed loop steering of unicycle like vehicles via lyapunov techniques. *IEEE Robot. Autom.*, 2(1), 27–35.

Bao, Q.Y., Li, S.M., Shang, W.Y., and An, M.J. (2009). A fuzzy behavior-based architecture for mobile robot navigation in unknown environments. In *Conf. Rec. IEEE/AICI*, volume 2, 257–261.

Baumann, C., Birch, H., and Martinoli, A. (2022). Leveraging multi-level modelling to automatically design behavioral arbitrators in robotic controllers. In *Conf. Rec. IEEE/IROS*, 9318–9325.

Benbouabdallah, K. and Zhu, Q.D. (2013). A behavior-based controller for a mobile robot tracking a moving target in multi-obstacles environment. In *Conf. Rec. IEEE/IHMSC*, volume 2, 418–423.

Bozzi, A., Graffione, S., Sacile, R., and Zero, E. (2022). Design and implementation of an asynchronous finite state controller for wheeled mobile robots. In *Actuators*, volume 11, 330. MDPI. doi:10.3390/act11110330.

Brooks, R. (1986). A robust layered control system for a mobile robot. *IEEE Robot. Autom.*, 2(1), 14–23.

Burjorjee, K. (2007). Speedyga: a fast simple genetic algorithm. *Mathworks Central*.

Cherroun, L. and Boumehraz, M. (2012). Intelligent systems based on reinforcement learning and fuzzy logic approaches, "application to mobile robotic". In *International Conference on Information Technology and e-Services*, 1–6. doi:10.1109/ICITeS.2012.6216661.

Colledanchise, M. and Ögren, P. (2018). *Behavior trees in robotics and AI: An introduction*. CRC Press.

Coumans, E. (2015). Bullet physics simulation. In *ACM SIGGRAPH Courses*, 1. Association for Computing Machinery.

De Silva, L. and Ekanayake, H. (2008). Behavior-based robotics and the reactive paradigm a survey. In *11th International Conference on Computer and Information Technology*, 36–43. doi:10.1109/ICCITECHN.2008.4803107.

Dhaouadi, R. and Hatab, A.A. (2013). Dynamic modelling of differential-drive mobile robots using lagrange and newton-euler methodologies: A unified framework. *Advances in Robotics & Automation*, 2(2), 1–7.

Horiuchi, Y. and Noborio, H. (2001). Evaluation of path length made in sensor-based path-planning with the alternative following. In *Conf. Rec. IEEE/ICRA*, volume 2, 1728–1735.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Rob. Res.*, 5(1), 90–98.

Kim, Y.H., Son, W.S., Park, J.B., and Yoon, T.S. (2016). Smooth path planning by fusion of artificial potential field method and collision cone approach. In *MATEC Web of Conferences*, volume 75, 05004. EDP Sciences.

Kubo, R., Fujii, Y., and Nakamura, H. (2020). Control lyapunov function design for trajectory tracking problems of wheeled mobile robot. *IFAC-PapersOnLine*, 53(2), 6177–6182.

Kuo, C.H. et al. (2013). Development of a fuzzy logic wall following controller for steering mobile robots. In *Conf. Rec. iFUZZY*, 7–12. IEEE.

Lynch, K.M. and Park, F.C. (2017). *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press.

Panahandeh, P., Alipour, K., Tarvirdizadeh, B., and Hadi, A. (2019). A kinematic lyapunov-based controller to posture stabilization of wheeled mobile robots. *Mechanical Systems and Signal Processing*, 134, 106319. doi:10.1016/j.ymssp.2019.106319.

Petrovic, P. (2008). *Evolving behavior coordination for mobile robots using distributed finite-state automata*. INTECH Open Access Publisher.

Ramakrishna Pandian, K.K., Dinh, T.Q., and Marco, J. (2021). Modelling and brake blending control for multi-drive mode electric two-wheelers. In *24th International Conference on Mechatronics Technology (ICMT)*, 1–6. doi:10.1109/ICMT53429.2021.9687134.

Ravangard, M. (2015). Fuzzy behavior based mobile robot navigation. In *Conf. Rec. 4th CFIS*, 1–7. doi:10.1109/CFIS.2015.7391649.

Robotics, A. (2006). Pioneer 3 operations manual.

Rohmer, E., Singh, S.P., and Freese, M. (2013). V-rep: A versatile and scalable robot simulation framework. In *Conf. Rec. IEEE/IROS*, 1321–1326.

Sathiya, V. and Chinnadurai, M. (2019). Evolutionary algorithms-based multi-objective optimal mobile robot trajectory planning. *Robotica*, 37(8), 1363–1382.

Shen, X. (2013). A quantum evolutionary algorithm for robot path planning in dynamic environment. In *Conf. Rec. IEEE/CCC*, 8061–8065.

Sutton, R.S. and Barto, A.G. (2018). *Reinforcement learning: An introduction*. MIT press.

Van Nguyen, T.T., Phung, M.D., and Tran, Q.V. (2017a). Behavior-based navigation of mobile robot in unknown environments using fuzzy logic and multi-objective optimization. *arXiv:1703.03161*.

Van Nguyen, T.T., Phung, M.D., and Tran, Q.V. (2017b). Behavior-based navigation of mobile robot in unknown environments using fuzzy logic and multi-objective optimization. *International Journal of Control and Automation*, 10(2), 349–364.

Ye, C. and Webb, P. (2009). A sub goal seeking approach for reactive navigation in complex unknown environments. *Robotics and Autonomous Systems*, 57(9), 877–888.