







Article

Fuzzy-Based Fault-Tolerant Control for Omnidirectional Mobile Robot

Ahmad M. Alshorman ^{1,*} , Omar Alshorman ² , Muhammad Irfan ³ , Adam Glowacz ^{4,*} ,
Fazal Muhammad ⁵  and Wahyu Caesarendra ⁶ 

¹ Mechanical Engineering, Mechanical Engineering Department, Jordan University of Science and Technology, Irbid 22110, Jordan

² Electrical Engineering Department, Faculty of Engineering, Najran University, Najran 61441, Saudi Arabia; omar2007_ahu@yahoo.com

³ College of Engineering, Electrical Engineering Department, Najran University, Najran 61441, Saudi Arabia; miditta@nu.edu.sa

⁴ Department of Automatic Control and Robotics, AGH University of Science and Technology, 30-059 Kraków, Poland

⁵ Department of Electrical Engineering, City University of Science and Information Technology Peshawar, Peshawar 25000, Pakistan; fazal.muhammad@cusit.edu.pk

⁶ Faculty of Integrated Technologies, Universiti Brunei Darussalam, Jalan Tungku Link, Gadong BE1410, Brunei; wahyu.caesarendra@ubd.edu.bn

* Correspondence: amalshorman6@just.edu.jo (A.M.A.); adglow@agh.edu.pl (A.G.)

Received: 5 August 2020; Accepted: 5 September 2020; Published: 9 September 2020



Abstract: The motion-planning problem is well known in robotics; it aims to find a free-obstacle path from a starting point to a destination. To make use of actuation generosity and the fuzzy fast response behavior compared to other non-linear controllers, a fuzzy-based fault-tolerant control for an omnidirectional mobile robot with four Mecanum wheels is proposed. The objective is to provide the robot with an online scheme to control the robot motion while moving toward the final destination with avoiding obstacles in its environment and providing an adaptive solution for a combination of one or combination of the wheel's faults. The faults happen when the wheel does not receive the control command signal from the controller; in this case, the robot can rotate freely due to the interaction with the ground. The principle of fuzzy-based control proposed by Sugeno is used to develop the motion controller. The motion controller consists of two main controllers: the Run-To-Goal, and the obstacle-avoidance controller. The outputs of these two controllers are superposed to get the net potential force on the robot. By its simplicity, the fuzzy controller can be suitable for online applications (online path planning in our case). To the best of our knowledge, this is the first fuzzy-based fault-tolerant controller for an omnidirectional robot. The proposed controller is tested by a set of simulation scenarios to check the proposed fuzzy tolerant control. Kuka OmniRob is used as an example of the omnidirectional robot in these simulation runs. Matlab is used to build the fuzzy-based fault-tolerant control, and the 3D simulation is developed on the CoppeliaSim software. We examine five distinct scenarios, each one with a different fault state. In all scenarios, the proposed algorithm could control the robot to reach its final destination with the absence and presence of an obstacle in the workspace, despite actuator faults, without crossing the workspace boundaries.

Keywords: mobile robots; path planning; fault control; fault identification; artificial potential field; fuzzy control; omnidirectional robot

1. Introduction

During research in recent decades, industry has investigated the applications of mobile robots in medicine, automotive, military, search and rescue, agricultural, services, underwater research, personal and other many applications in real life [1–8]. In any of these pivotal applications, the robot needs to interact with the environment. Also, it is required to do their tasks without or with minimum human input and control. As a result, it is desired to complete their missions and reach the desired destination (goal). This problem is well known in the field of robotics as a motion-planning and control problem. Many works on the topic have been investigated; the objective focuses on generating a collision-free path from the current state of the robot to the final destination. Obstacle avoidance is considered one of the basic requirements for the robot to do the task efficiently. In 1985, Oussama Khatib proposed a new method for real-time obstacle avoidance for mobile robots based on the artificial potential field. The potential field considers the robot's environment as a field of forces. The goal destination attracts the robot (attractive force); in contrast, the obstacles are considered repulsive forces that push the robot away. The resultant force is the sum of attractive and repulsive forces. They extended the method for moving objects by using time-varying artificial field methods [9,10]. In [11], Zavlangas et al. proposed a local, real-time fuzzy-based approach for navigation and obstacle avoidance for omnidirectional mobile robots. They consider the nearest obstacle strategy in the fuzzy-based path planning algorithm. The algorithm considers the distance, location to the nearest obstacle, and the robot's current position and direction and destination.

In [12], Ren et al. presented a fuzzy-based intelligent obstacle-avoidance strategy for the wheeled mobile robot. The robot controller consists of two main fuzzy logic controllers. When there are no obstacles in the robot's path, the Run-to-Goal controller is responsible for making the robot approach the desired goal. In the presence of any obstacle, while it is heading to the goal, another fuzzy-based obstacle controller will generate commands to the robot to avoid that obstacle. As a more potent tool, in 2019, Nadour et al. presented a hybrid type-2 fuzzy controller and optical flow approach using the Horn–Shunk algorithm. They used the Takagi–Sugeno fuzzy controller to avoid the detected obstacle based on another image processing algorithm. The obstacle and its position are defined in the image processing using the Horn–Shunk algorithm. The results show the effectiveness of the visual-based navigation system [13].

In [14], the authors introduced a real-time optimal path planning of humanoid robots. They discussed the problem of the unknown environment with the presence of dangerous space. They defined the danger space where the robot is permitted to go when no free space available to go through. In other words, the robots cannot cross the danger space once at least one free path is available. Mud pits and greasy areas are examples of these dangerous areas. They also used a modified Markov decision method based on the Takagi–Sugeno fuzzy system to achieve the task (reach the destination in the presence of danger area).

Since it has wide applications and use, autonomous mobile robots must work with reliable performance and increased autonomy. The type of wheel, the configuration of the autonomous robot, and the actuators' availability determine the robot's capabilities and ability to maneuver in its environment. When the robot moves in an environment with N physical dimensions, and it has geometrical constraints on due to the interaction with the ground, then this type of robots called a non-holonomic, and in this case, the robot has a set of geometrical constraints and cannot move in all physical dimensions. In other words, the robot has fewer degrees of freedom (DOF) than the total available due to the constraints on the robot. Car-like robots, Unicycle, and differential drive robots are examples of non-holonomic robots.

In contrast, the holonomic robots can move in any direction (with N dimensions) of its space [15,16]. In these types of robots, the robot can be controlled to move in any direction since it has the same controllable directions as the same as the total DOF. The robot can move and rotate in any directions freely without any geometrical constraints, which make these types of robots at the center of the attention. The omnidirectional wheel mobile robot is an example of a holonomic

mobile robot. In this type of ground robots, the robot can move or rotate in any direction based on the actuators' commands.

In most cases, designers and researchers build a navigation algorithm and assume the robot working with all its functions. In real applications, this is not the case, and the robot may suffer from faults in one or more of its actuators or/and sensors. For the robot to work efficiently and reach its goals in these cases, it must express and show tolerant behavior [15,17–19]. The fault diagnosis method is known as fault detection and identification (FDI), the system's actual response is compared to the response of the theoretical response for the same input signal, if there is a deviation between these responses, then the fault is detected [20].

After identifying the fault, the next step is to impart fault-tolerant control that retains the system's performance and behavior, the closest possible to the ideal situation just before the fault. In literature, there are two main fault-tolerant control techniques used in such cases. The first one is the passive approach, in which the fault is considered a perturbation to the system. The second approach is the active method, where the pre-defined or new online control algorithms are used after identifying the system's faults. This pre-defined control algorithm depends on the type of failure in the systems; it may occur in one part, or a combination between more than one failure in the actuators or/and the sensors [15,16,21–23].

In [24], the authors presented the kinematics and a method for fault-tolerant control of a three-wheel omnidirectional soccer robot, to deal with the sudden failure of one motor during the match. Ref. [23] proposed a novel gear train omnidirectional mechanism for precise and high robot mobility. In this paper, the fault tolerance problem of the omnidirectional wheels was dealt with the gear train's mechanical design. The mechanism uses three motors with conventional tire wheels. When one of the three motors are not working correctly, the robot motion is controlled by the other two healthy motors; in this case, the robot's structure and movement becomes similar to a differential two-wheeled mobile robot subjected to non-holonomic constraints [23]. Chao Ren et al. presented a reduced-order extended state observer-based sliding mode control for a three omnidirectional robot with friction compensation [25].

Qi Song et al. proposed a fault-tolerant control based on a three-degree omnidirectional robot's inverse dynamics. The inverse dynamic control is enhanced by actuator effectiveness factors introduced to the omnidirectional robot's dynamic equation. Unscented Kalman filter is used to estimate the robot's state and the actuator effectiveness factors [26].

The paper [27] represents an analysis of the four Mecanum-wheeled omnidirectional robot and performance analysis of trajectory tracking in case of one fault and two faults in the wheel's motors. The robot gives the desired performance in one fault case, and the system remains in its full actuation capabilities. The performance was not the same for the two motors' faults situation, where some of these fault combinations do not give the desired performance while some do.

Damiano Rotondo et al. proposed trajectory tracking control of a four-wheeled omnidirectional mobile robot based on the reference model approach. The control loop was enhanced by adding a switching quasi-Linear Parameter Varying to obtain fault tolerance. A set of simulations and experiments were performed to check the performance of the proposed method [28]. The work in [29] studies the FDI of actuation and sensing of an omnidirectional mobile robot with four Mecanum wheels. The approach is based on residual where the isolation of sensors based on the analyzing residual signatures, which are different for each sensor fault. For actuators faults more residual characteristics are taken into consideration to achieve the isolation.

In [30] Damiano Rotondo et al. present fault-tolerant control of an omnidirectional robot using a Switched Takagi–Sugeno approach strategy. To achieve the desired performance and keep the stability, they rely on their adaptive controller solution. To prove the effectiveness of the proposed controller, experiential results obtained for a four-wheeled omnidirectional mobile robot. Based on the experimental results that showed potential and performance, the proposed control approach was able to recover the desired trajectory tracking when the first wheel motor was subjected to a loss of

effectiveness fault. All the mentioned FTC schemes deal with trajectory control, and no one discusses this problem in the presence of obstacles in the robot's path. Ref. [15] presents the design of an FTC for an omnidirectional robot with four Mecanum wheels. In this research, a non-linear predictive controller is proposed to solve multiple combinations of actuation faults problem. Depending on the identified fault, the control vector and the dynamic of the robot are reconstructed.

In this paper, we proposed a fuzzy-based tolerant control strategy for a four-wheel Mecanum omnidirectional mobile robot which appropriately uses the robot's actuation redundancy, so this will afford adaptive control solutions in the case of failure of one actuator or more of the robot. The proposed navigation and control strategy is based on Takagi–Sugeno Fuzzy Model and is composed of two fuzzy logic controllers. The first one is the Run-to-Goal controller which will control the robot to go to the final goal. In detecting an obstacle in the robot path, the second obstacle-avoidance controller will take the robot away from the obstacle, which depends on the position and the directions of both the obstacle and the final destination relative to the robot's orientation and position. These two controllers and the fuzzy rules and output to the actuators are adaptive and modified based on their health condition. We consider actuation faults where the wheels can rotate freely due to the friction with the ground but cannot receive the controller's control signal.

This paper's main contribution is to design an adaptive fault-based controller with a fast response and suitable for an online application scheme to control the omnidirectional robot with four Mecanum wheels robot motion. Consequently, this controller can control the robot to reach its final destination with the absence and presence of an obstacle in the workspace, which lacks most literary works, despite the existence of actuator faults, without crossing the workspace boundaries.

2. Kinematics and Dynamics of the 4 Mecanum Omnidirectional Wheel Mobile Robot

In this paper, we assume the robot as a rigid body on 4-mecanum wheels omnidirectional robot. The model assumes that the robot operates on a horizontal plane, and it considers that there is no slippage on the robot's wheels. Also, we assume that the robot's geometric parameters are known as well as the dynamic parameters. As a result, the robot chassis has a three-dimensionality, two for the plane's position, and the third dimension for the robot's orientation around the orthogonal axes to the plane of motion. Referring to Figure 1, the global reference frame is defined by the two axes X_I and Y_I from the origin $O : \{X_I, Y_I\}$. To determine the robot's position, we choose the point P on the robot's center as a reference point. The robot's body reference frame is defined by the two axes $\{X_R, Y_R\}$ relative to the robot's center P . Thus, the robot poses ξ_I can be described using the position of the point P relative to the inertial reference frame I , and the angular rotation of the robot θ relative to the I .

$$\xi_I = \begin{pmatrix} x_p \\ y_p \\ \theta \end{pmatrix} \quad (1)$$

To describe the robot motion with respect to the I :

$$\dot{\xi}_I = R_\theta \cdot \dot{\xi}_R \quad (2)$$

where $R_\theta = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$ is an orthogonal 7 the rotational matrix used to map the motion in the local reference frame $\{X_R, Y_R\}$ to the global reference frame $\{X_I, Y_I\}$. Also, $\dot{\xi}_R$ is the body velocity vector of the robot in the local reference frame and given by $\dot{\xi}_R = [u, v, \omega]^T$. Since the robot has four wheels and each wheel has a radius r , and rotate with a rotational velocity ω_i , then we can relate the body velocity vector $\dot{\xi}_R$ to the wheels rotational velocities vector $\omega = [\omega_1, \omega_2, \omega_3, \omega_4]$, using the following expression:

$$\dot{\xi}_R = J_v \omega \quad (3)$$

Assuming that the offset of roller for the omnidirectional wheels used in this research is 45° . Then the Jacobean matrix J_v can be written as(see Figure 1):

$$J_v = \frac{1}{4}r \begin{pmatrix} 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ \frac{1}{a+b} & \frac{1}{a+b} & \frac{1}{a+b} & \frac{1}{a+b} \end{pmatrix} \tag{4}$$

where a, b are the distance from the wheel center to X_R and X_Y , respectively, as shown in Figure 1.

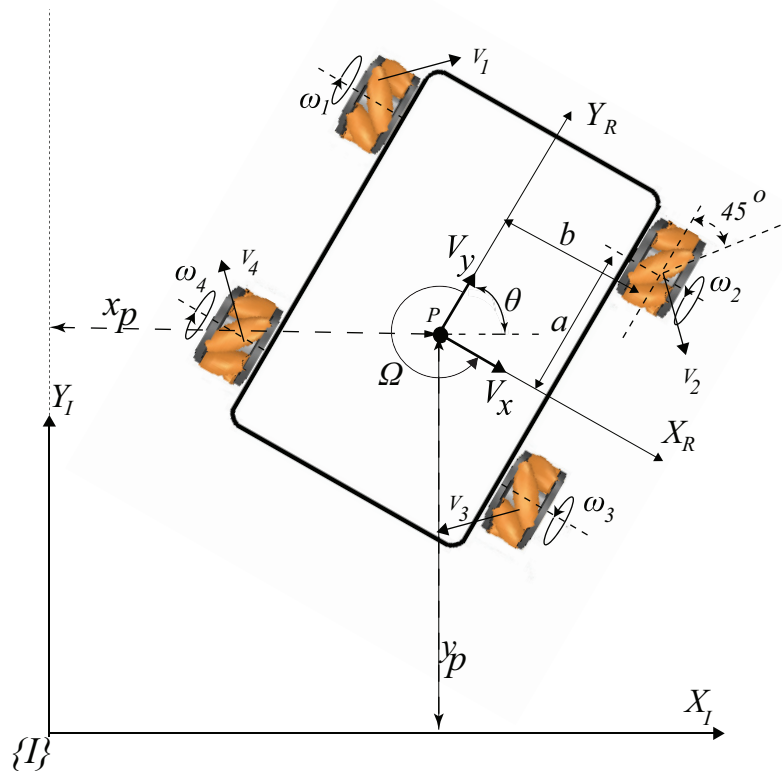


Figure 1. Kinematic representation of the Omni-direction wheels. And the global reference frame and the robot local reference frame.

The dynamics of the robot can be written in a matrix form using Lagrange:

$$M\dot{\omega} + D_\omega\omega = \tau \tag{5}$$

where τ is the actuators torque vector, $\tau = [\tau_1, \tau_2, \tau_3, \tau_4]^T$, and τ_i is the torque of the motor actuates the i th wheel, and $i = \{1, 2, 3, 4\}$. ω and $\dot{\omega}$ are the rotational vectors of the rotational velocity and acceleration receptively. M is a square symmetrical mass matrix and is given by:

$$M = \begin{pmatrix} \frac{r^2 I_z}{16(a+b)} + \frac{mr^2}{8} + I_\omega & -\frac{r^2 I_z}{16(a+b)} & \frac{r^2 I_z}{16(a+b)} & \frac{mr^2}{8} - \frac{r^2 I_z}{16(a+b)} \\ -\frac{r^2 I_z}{16(a+b)} & \frac{r^2 I_z}{16(a+b)} + \frac{mr^2}{8} + I_\omega & \frac{mr^2}{8} - \frac{r^2 I_z}{16(a+b)} & \frac{r^2 I_z}{16(a+b)} \\ \frac{r^2 I_z}{16(a+b)} & \frac{mr^2}{8} - \frac{r^2 I_z}{16(a+b)} & \frac{r^2 I_z}{16(a+b)} + \frac{mr^2}{8} + I_\omega & -\frac{r^2 I_z}{16(a+b)} \\ \frac{mr^2}{8} - \frac{r^2 I_z}{16(a+b)} & \frac{r^2 I_z}{16(a+b)} & -\frac{r^2 I_z}{16(a+b)} & \frac{r^2 I_z}{16(a+b)} + \frac{mr^2}{8} + I_\omega \end{pmatrix} \tag{6}$$

where m is the total mass of the robot with the wheels, also, I_z , and I_ω are the moment of inertia for the robot and the wheels, respectively. In Equation (5), D_ω is a diagonal matrix and denotes the friction coefficient between the wheels and the grounds, if we assume that all wheels have the same

coefficient of friction with the ground all the time (μ_k), then D_ω is given by $D_\omega = \mu_k \mathbf{I}_{4 \times 4}$. More details about the kinematic and dynamic of the omnidirectional robot with 4-mecanum wheels can be found in [15,16,31–33]

3. Problem Statements

We consider an omnidirectional robot with four Mecanum wheels. We assume that the robot moves on a flat terrain surface and can move to any point in its workspace other than the obstacles area. Also, we assume that the robot moves with no or with little slipping. All the geometric and dynamic parameters of the robot are known. The robot's position, velocity, and pose and the torque of motors wheel (current of the motor) are measurable all the time.

We aim to design a fuzzy-based controller to control the robot to reach the final destination with the absence and presence of an obstacle in the workspace despite the existence of actuator faults. Figure 2 shows the simulation environment, where we have four static obstacles and the final destination as well as a KUKA Omnirob (the omnidirectional robot). In this paper, we consider actuation faults that can occur in one or a combination of fault on the wheels. The wheel failure occurs when it can rotate freely due to the friction with the ground but cannot receive the controller's control signal (there is no control on the faulty motor wheel). Thus, we proposed a fuzzy logic tolerant control strategy for an omnidirectional mobile robot with four-wheel Mecanum, which appropriately uses the robot's actuation redundancy. The control strategy will afford adaptive control solutions in case of failure, where the proposed navigation and control strategy is based on Takagi–Sugeno. The fuzzy controller is considered suitable for online applications because of its simplicity and fast time response. It is composed of two fuzzy logic controllers; the first one is the Run-to-Goal controller which will control the robot to reach the final goal. The second controller works in case of detecting an obstacle in the robot path. The second obstacle-avoidance controller will take the robot away from the obstacle; this depends on the position and the directions of both the obstacle and the final destination relative to the robot's orientation and position. These two controllers and the fuzzy rules, as well as the control signal to the actuators, are adaptive and modified based on the health condition of the actuators.

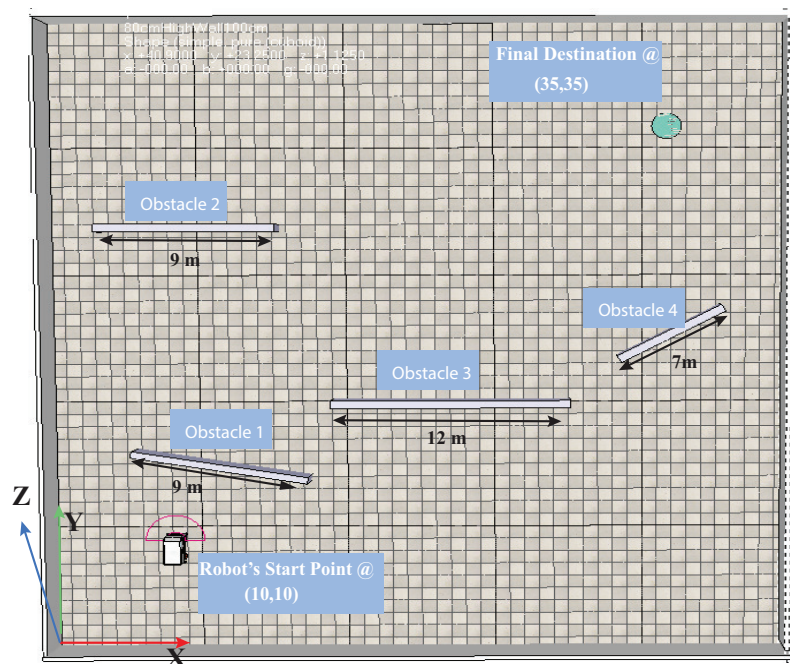


Figure 2. Simulation Environment.

Figure 3 shows the flow chart of the main steps and proposed algorithm of path planning and control. The robot starts by setting the robot's orientation toward the goal, and then it starts

moving toward the robot using the Run-to-Goal controller. If any obstacle popped up, then the Obstacle-Avoidance controller is activated. These two controllers are modified based on the robot's dynamics, which may be changed during the travel time if any failure is defined. For example, while the robot is heading to the goal and the front motor's wheel is identified to be faulty. As a result, the control signal to the rear wheels will be modified to maintain the robot's efficiency and complete the mission even there are one or more faulty wheels.

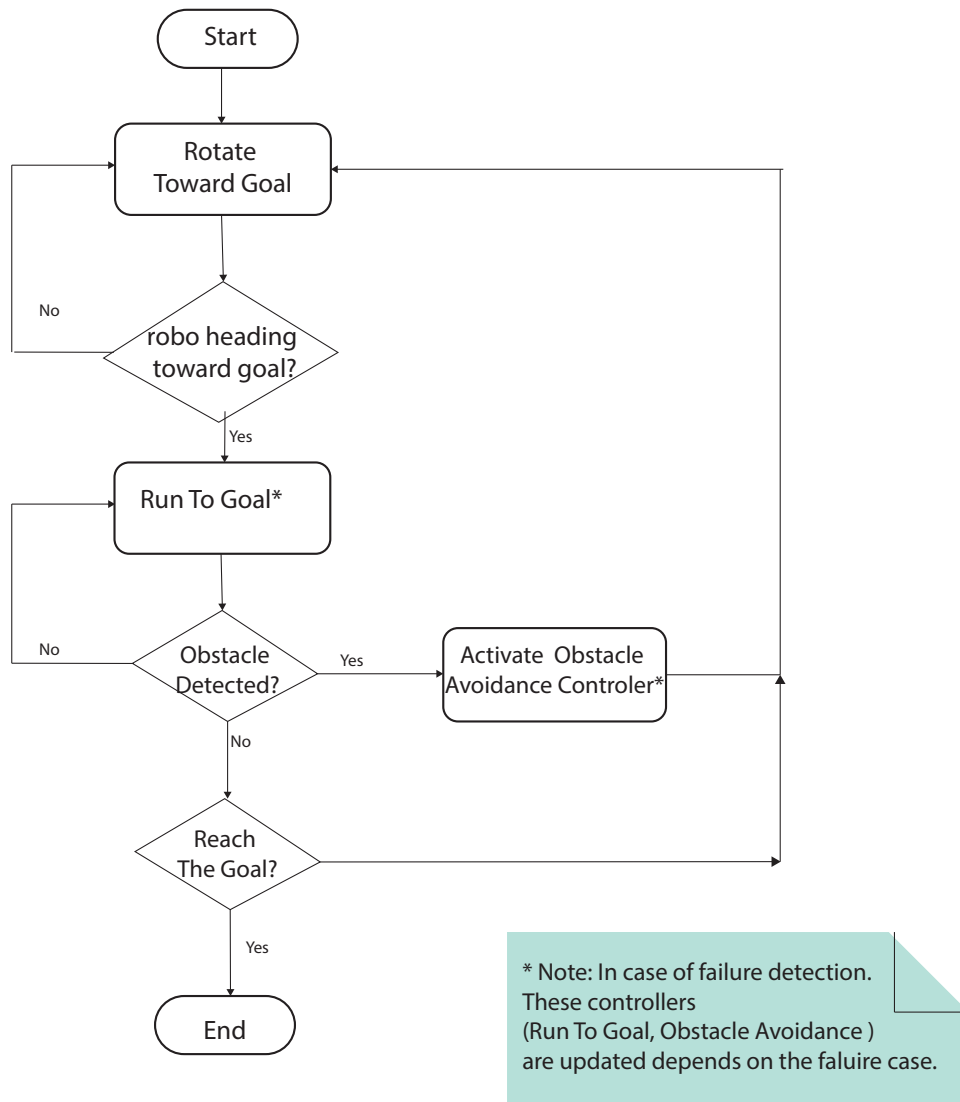


Figure 3. Flow chart for the path planning algorithm.

4. Fault Detection, Identification, and Isolation

In this section, we will describe the procedure of fault detection and identification. We assume the wheel is faulty if it cannot receive the command signal from its controller, but it is still rotating freely. We follow the same procedure proposed by [15] to diagnose the malfunction of each motor of the robot. Each motor's torque is proportional to the armature current only if we assume that the magnetic field is constant. The relationship between the torque of the motor τ_m and its armature current i is given by:

$$\tau_m = Ki \quad (7)$$

where K is a constant.

Thus, the residuals between the measured torque and the calculated torque from the measured current are given by :

$$\begin{aligned} r_1 &= \tau_{m1} - Ki_1 \\ r_2 &= \tau_{m2} - Ki_2 \\ r_3 &= \tau_{m3} - Ki_3 \\ r_4 &= \tau_{m4} - Ki_4 \end{aligned} \quad (8)$$

Thus, the fault can occur when the residual value $r_i \geq r^*$ ($i = 1, 2, 3, 4$), where r^* is a threshold value for each motor residual. In other words, if the residual exceeds the r^* (it is a pre-defined value and depends on the motor), then the motor will be identified as a fault motor. Then, based on each motor's residual, the fault isolation can be clearly described by the fault detection and identification system. The fault may occur in any motor of the four motors wheel. Next, we discuss all possible states that may result from the fault in one or more motors wheel.

- **State1:** Where all motors are healthy motors (no faults) and the robot is fully operational. Thus, the control input signal $\tau_{control}$ is given by :

$$\tau_{Control} = [\tau_1, \tau_2, \tau_3, \tau_4]^T \quad (9)$$

In this case, the robot can be controlled using the four available motors.

- **State2:** In this state, only one motor of the four is identified as a faulty motor. Depending on which motor has failed, the control signal will form. As a result, the control signal may be given by the following [15]:

$$\tau_{Control} = \begin{cases} [\tau_1, \tau_2, \tau_3]^T & \text{if } r_4 \geq r^* \\ [\tau_1, \tau_2, \tau_4]^T & \text{if } r_3 \geq r^* \\ [\tau_1, \tau_3, \tau_4]^T & \text{if } r_2 \geq r^* \\ [\tau_2, \tau_3, \tau_4]^T & \text{if } r_1 \geq r^* \end{cases} \quad (10)$$

- **State3:** In this case, either the front wheels (1 & 2) or the rear wheels (3 & 4) are identified simultaneously as faulty wheels. In this case, the control signal is given by:

$$\tau_{Control} = \begin{cases} [\tau_1, \tau_2]^T & \text{if } r_3 \geq r^* \ \& \ r_4 \geq r^* \\ [\tau_3, \tau_4]^T & \text{if } r_1 \geq r^* \ \& \ r_2 \geq r^* \end{cases} \quad (11)$$

- **State4:** In this case, either the right wheels (2 & 3) or the left wheels (1 & 4) are identified simultaneously as faulty wheels. In this case, the control signal is given by:

$$\tau_{Control} = \begin{cases} [\tau_2, \tau_3]^T & \text{if } r_1 \geq r^* \ \& \ r_4 \geq r^* \\ [\tau_1, \tau_4]^T & \text{if } r_2 \geq r^* \ \& \ r_3 \geq r^* \end{cases} \quad (12)$$

- **State5:** In this case, the faults are in the diagonal set of wheels. Either the right wheels (1 & 3) or the wheels (2 & 4) are identified simultaneously as faulty wheels. In this case, the control signal is given by:

$$\tau_{Control} = \begin{cases} [\tau_2, \tau_3]^T & \text{if } r_1 \geq r^* \ \& \ r_4 \geq r^* \\ [\tau_1, \tau_4]^T & \text{if } r_2 \geq r^* \ \& \ r_3 \geq r^* \end{cases} \quad (13)$$

- **State6:** In this case, more than two actuators are failed simultaneously. Thus, in this case, the robot is uncontrollable and cannot complete the mission of the robot.

In the first two states, 1 and 2, the robot is fully actuated and can be moved anywhere in any direction. Although the robot is underactuated in the states 3, 4 and 5 still, it can be moved and controlled to do the mission. The robot path planning control consists of two main problems [11,34]:

- **Run – To – Goal** : Run-To-Goa is considered a global problem because local information cannot help find the optimal short path to the goal. Then, knowing the topology of the space is crucial to achieving optimal planning. In this research, we use the information about the final distention and the pose of the robot and its location relative to its goal (final destination) to generate the attractive force to the goal, as discussed in Section 5.
- **Obstacle Avoidance** : Only local information (sensors data) are required to solve the obstacle-avoidance problem. These problems cannot be solved in advance in case of a dynamic environment, where the obstacles and the environment's components are moving. In the existence of an obstacle, the position of the robot 5.

In this research, the above two problems are modified according to the health state discussed. See Figure 3.

5. Fuzzy Fault-Tolerant Control

In this section, we discuss the fault-tolerant control strategy. We were influenced by the artificial potential field method proposed by Khatib [9]. In his research, Khatib uses the potential filed concept to compute the total force acting on the robot. This force has two components: the repulsive force in the existence of the obstacle, and the attractive force produced by the target to attract the robot. The potential field force is created by superposing these two forces, and the only nearest obstacle is considered to reduce the computational cost. Thus, the actuation control signal for each motor is given by:

$$\tau_i = \Gamma(F_{att} + \mu F_{rep}) \quad (14)$$

where Γ is a $n \times 1$ vector with a single nonzero (*equals one*) component defines the control vector components ($\tau_{Control}$) based on the states of faults discussed in Section 4. As an example, if the FDI algorithm identified the *state3* fault (front or left wheels have faults), then $\Gamma = [0, 0, 1, 0, \dots, 0]_{n \times 1}^T$. Each fault state has one or more fault cases; in state 3 (diagonal fault) has two cases: either the wheels 1 and 3, or the wheels 1 and 3 are faulty (both of these two cases belong to the state 3). Thus, the total number of fault cases in the five states is defined by n (11 in our problem). F_{att} and F_{rep} are vectors of the attractive and repulsive forces, respectively. Also, μ is a piecewise function (to activate the obstacle-avoidance controller, see Figure 3) and given by:

$$\mu = \begin{cases} \alpha & P(Obs) \leq d_o \\ 0 & P(Obs) > d_o \end{cases} \quad (15)$$

where α is a constant factor to be tuned, $P(Obs)$ is the position of the nearest obstacle to the robot (i.e., position sensors data), and d_o is the influence enrage of the obstacle, it can be adjusted based on the running speed of the robot and the workspace size (tuned parameter).

5.1. Fuzzy-Based Run-to-Goal Problem

The principle of fuzzy-based control proposed by Sugeno was used to develop the Run-To-Goal controller. In this approach we have two inputs: the distance from the current position of the robot to the goal D_G , and the angle between the robot direction θ and the angle of the straight line connecting the current position of the robot and the final goal position β , see Figure 4. The input space is divided by fuzzy memberships; a trapezoid and symmetrical triangle shape for the membership functions were used to the fact the computation, see Figure 5. We use nine-membership functions for the angle input θ ; this angle could be negative or positive depends on the position of the final goal and the robot's orientation. Also, we use four membership functions for the distance between the robot the final goal position. The output of this algorithm is considered to be an attractive force to the final target position. We use nine-membership crisp output values. Matlab Fuzzy logic toolbox was used to build the fuzzy controller, and the fuzzy rules were created based on the prior knowledge of the problem domain.

A total of 36 rules were generated to cover all possible inputs; Table 1 shows a sample of these fuzzy rules. When there is no obstacle in the path, the robot always tries to adjust the heading angle toward the target.

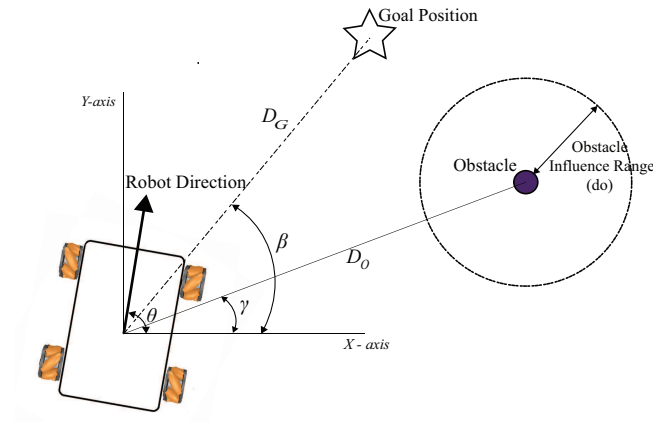
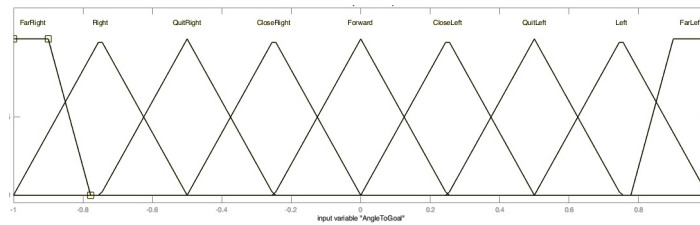
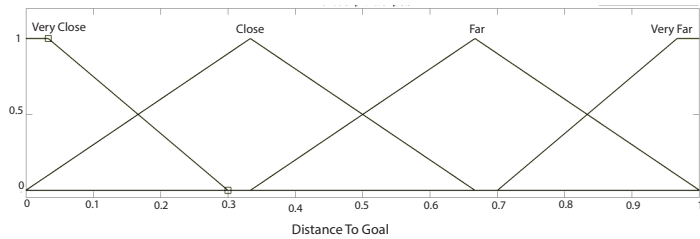


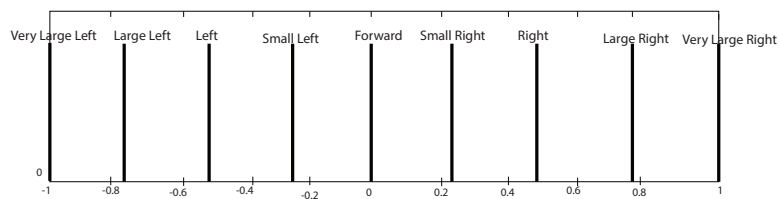
Figure 4. Robot Motion with Obstacle.



(a) Angle to The Goal, $\beta - \theta$.



(b) Distance to The Goal, D_G .



(c) Motor Signal Command: Output.

Figure 5. Fuzzy Membership Functions for the Run-To-Goal Controller.

Table 1. Run-To-Goal Controller Fuzzy Rules Sample.

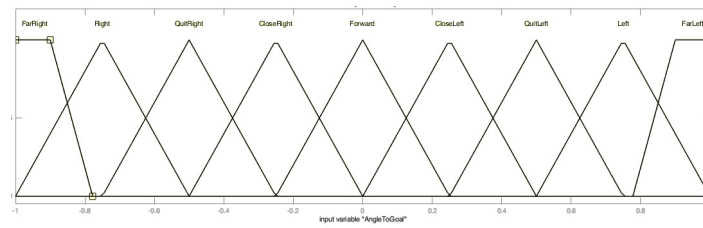
Angle to the Goal	Distance to the Goal	Motor Signal Command
HighRight	VeryClose	VeryLargeRight
Right	VeryClose	VeryLargeRight
QuitRight	VeryClose	VeryLargeRight
CloseRight	VeryClose	VeryLargeRight
HighRight	Close	VeryLargeRight
Right	Close	LargeRight
QuitRight	Close	Right
CloseRight	Far	SmallRight
Forward	Far	Forward
Forward	VeryClose	Forward
Left	VeryFar	SmallLeft
Left	VeryClose	VeryLargeLeft

5.2. Fuzzy-Based Obstacle Avoidance

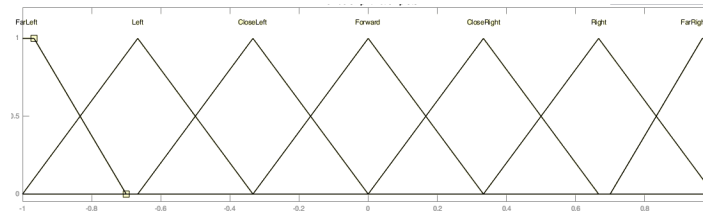
We use the fuzzy control principle by Sugeno to develop the obstacle-avoidance controller. The fuzzy model has three inputs:

- The distance from the current position of the robot to the nearest obstacle D_O . See Figure 4.
- The difference between the robot's orientation angle θ and the angle γ . See Figure 4.
- The difference between the robot's orientation angle θ and the angle β is shown in Figure 4.

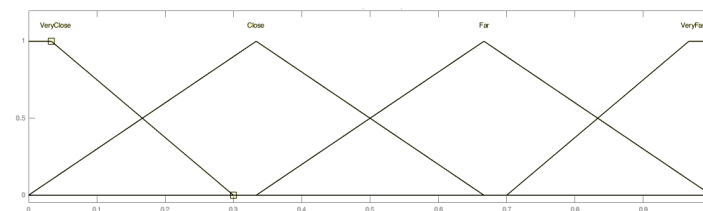
This input space is divided by fuzzy sets; trapezoid and symmetrical triangle shapes for the membership functions were used to reduce the computation efforts, see Figure 6. We use nine-membership functions for the angle input θ ; this angle could be negative or positive depends on the position of the final goal and the robot's orientation. Moreover, we use four membership functions for the distance between the robot and the nearest obstacle in its range, D_O . We also use seven membership functions for the angle between the robot orientation and the obstacle ($\beta - \gamma$). The output of this controller is considered to be a repulsive force. For the output, we use nine-membership crisp output values. Matlab Fuzzy logic toolbox was used to build the obstacle-avoidance fuzzy controller. Figure 7 shows the fuzzy surface of the rules for the obstacle-avoidance controller. A total of 324 rules were generated to cover all possible input combinations. Table 2 shows a sample of these fuzzy rules.



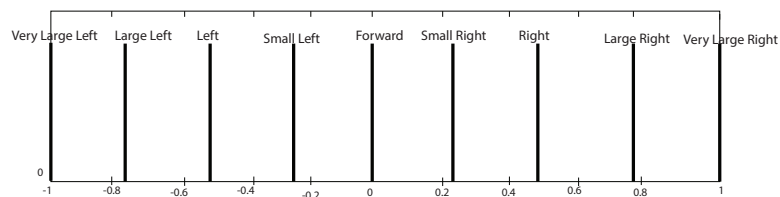
(a) Angle to The Goal, $\beta - \theta$.



(b) Angle to the Obstacle, $\beta - \gamma$.



(c) Distance to the Obstacle, D_O .



(d) Motor Signal Command: Output.

Figure 6. Fuzzy Membership Functions for the Obstacle-Avoidance Controller.

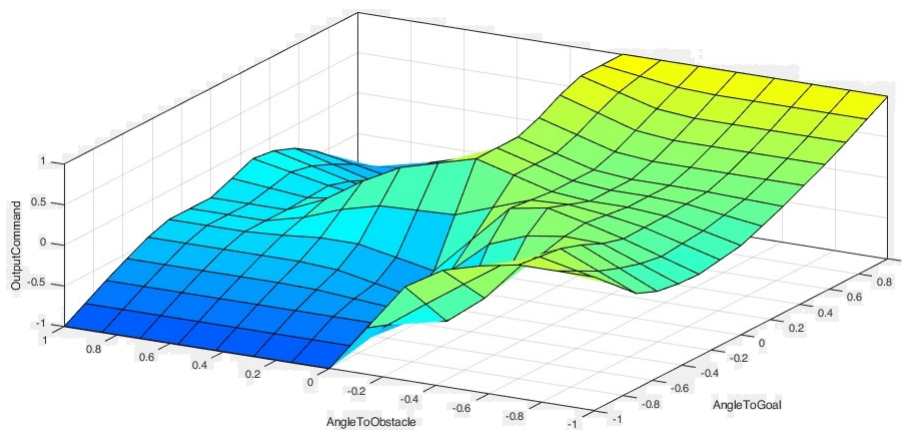


Figure 7. Fuzzy Surface of the Obstacle-Avoidance Controller.

Table 2. Obstacle-Avoidance Fuzzy Rules sample.

Angle to the Goal	Distance to Obstacle	Angle to Obstacle	Motor Signal Command
HighRight	VeryClose	HighRight	VeryLargeLeft
Right	VeryClose	Right	VeryLargeLeft
QuitRight	VeryClose	CloseRight	VeryLargeLeft
CloseRight	VeryClose	Forward	VeryLargeRight
HighRight	Close	HighRight	VeryLargeLeft
Right	Close	Left	LargeRight
QuitRight	Close	CloseLeft	Right
CloseRight	Far	Forward	SmallRight
Forward	Far	CloseRight	SmallLeft
Forward	VeryClose	Left	HighRight
Left	VeryFar	Right	SmallLeft
Left	VeryClose	Right	VeryLargeLeft

6. Simulation Results

In this section, a set of simulation scenarios is presented to check the proposed fuzzy tolerant controller's validity and performance in this paper. Matlab was used to build the fuzzy-based fault-tolerant control, and the 3D simulation was developed on the CoppeliaSim software. KUKA omniRob Model in CoppeliaSim was used as an example of the omnidirectional robot. We examine five distinct scenarios, each one with a different fault state. The goal is to reach the goal position while avoiding any obstacle in the path with respecting the workspace boundaries. Figure 2 shows the workspace and its boundaries, the robot, four walls (Obstacles), as well as the final goal position. The position of the final destination is {35 m, 35 m}. The CoppeliaSim simulation environment communicated with the Matlab using the remote API, which allows the communication and bidirectional data streaming between the CoppeliaSim and Matlab.

Figures 8 and 9 examine the performance of the proposed control strategy with no fault conditions (*State1*). In this scenario, all four motors are working properly. Figure 10 shows the robot trajectory, where the robot was able successfully to reach the final destination by avoiding the obstacle in its path. The robot could not reach the final destination with a straight line because of the obstacle in its direct path, but it could avoid the obstacle with minimal efforts needed.

Figures 10 and 11 examine the performance of the proposed control strategy with identified fault in the *wheel1* (*State2*). In this scenario, all motors work properly, but the first wheel (front-left) has a fault identified by the FDI algorithm. Figure 10 shows the robot trajectory where the robot was able successfully to reach the final destination by avoiding the obstacle in its path.

However, Figures 12 and 13 examine the performance of the proposed control strategy with faults on the Front-Right and Rare-Left wheels (Diagonal fault, *State3*). In this scenario, Only the Front-Left and Rear-Right wheels (1 and 4) can receive control signals. Figure 12 shows the robot trajectory where it was able successfully to reach the final destination with avoiding the obstacle in its path.

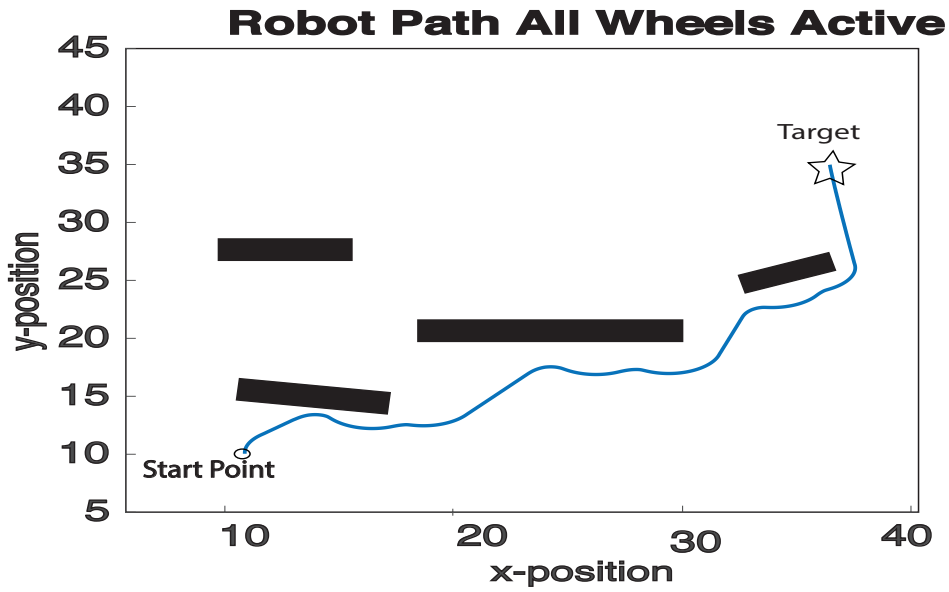


Figure 8. Path of the robot for state 1, All wheels working.

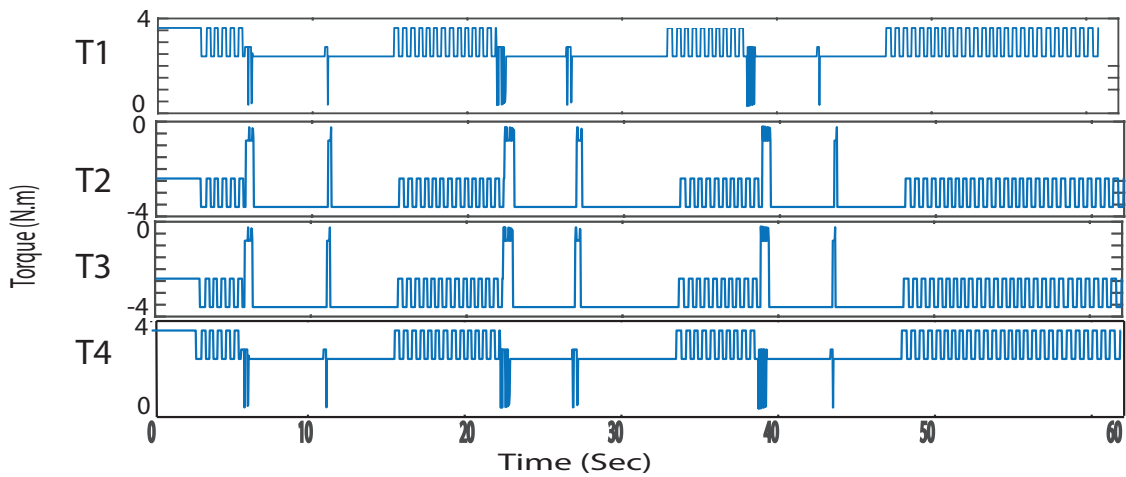


Figure 9. Simulation Results for State 1: Torque of motor 1,2,3,4 (All wheels working).

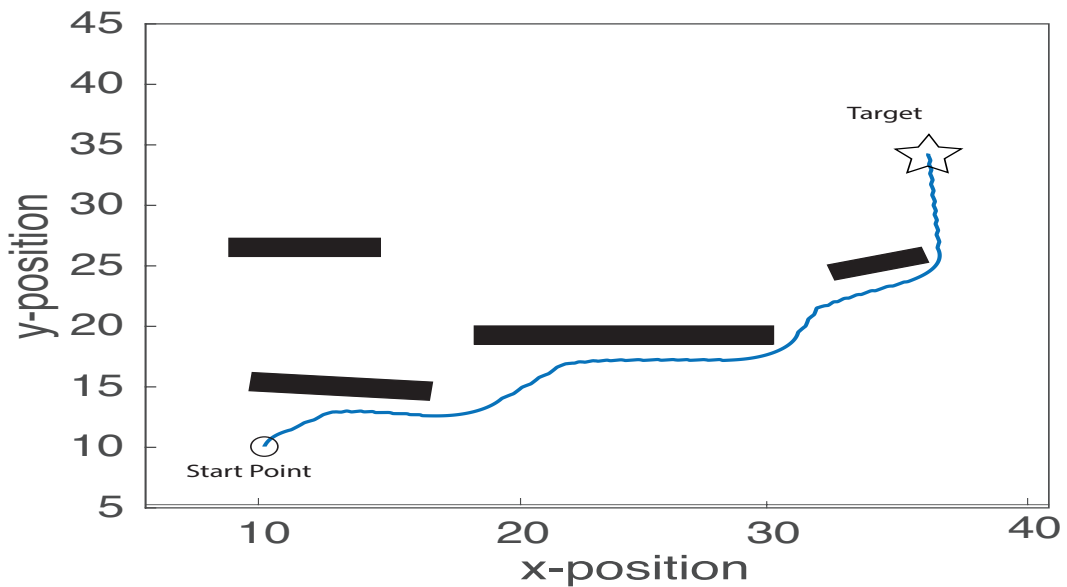


Figure 10. Path of the robot for state 2, the Front-Left motor is faulty.

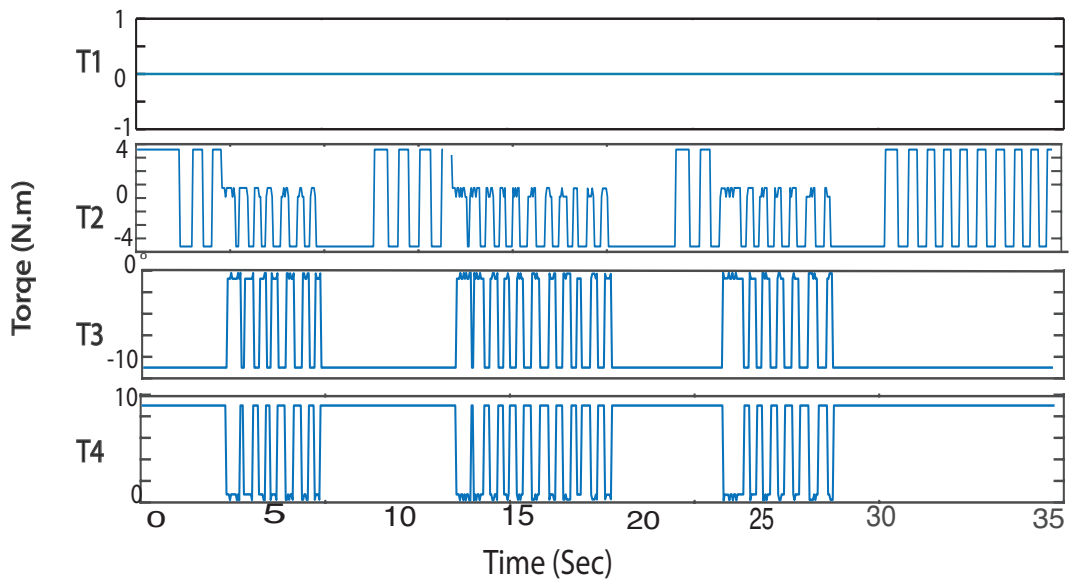


Figure 11. Simulation Results for State 2: Torque of motor 1,2,3,4 For state2.

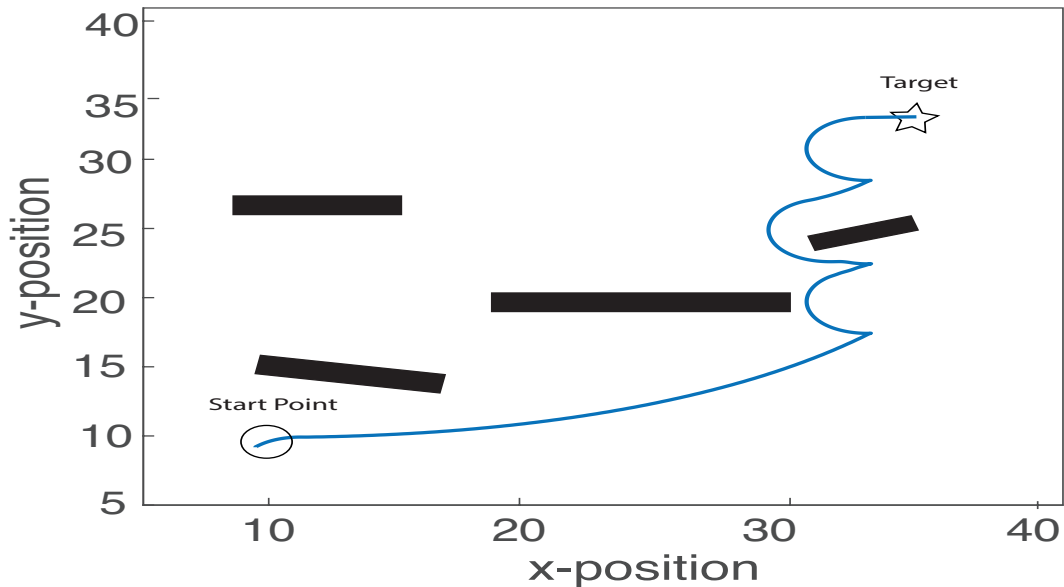


Figure 12. Path of the robot for state 3, state: Diagonal faults Right-Front and Left -Rear motors are faulty (Wheel 2 and 4).

Also, Figures 14 and 15 show that the robot can finish the desired task when the two right motors (2 , 3) are not functional. In this scenario, the right motors work probably, and due to the faults in the left motor, it can be seen that the robot reach the final destination, but it was not able to go straight to the goal because of the actuation faults, it takes some time (going right and left) to reach the goal.

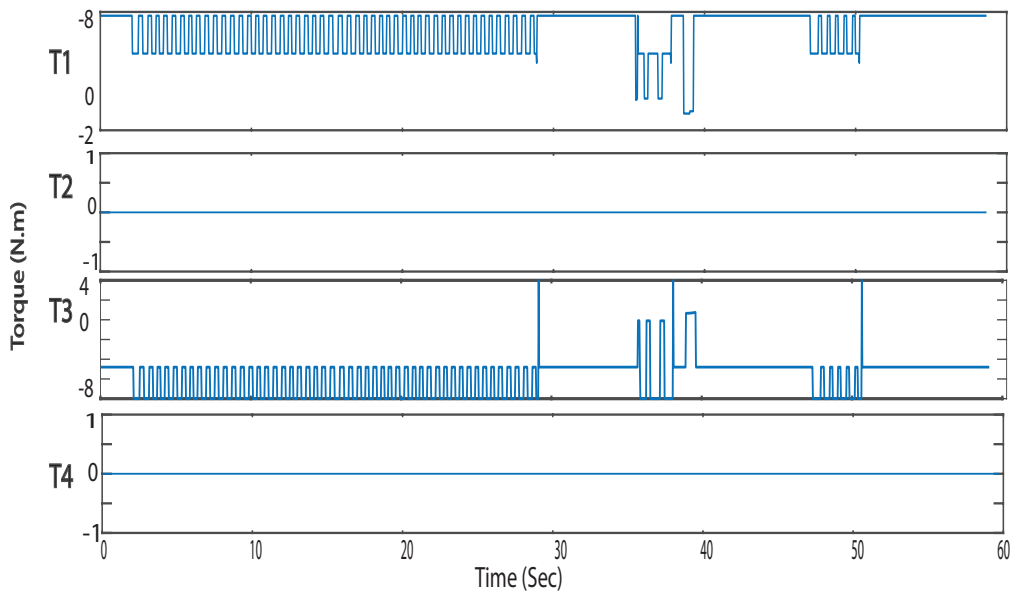


Figure 13. Simulation Results for State 3: Torque of motor 1,2,3,4.

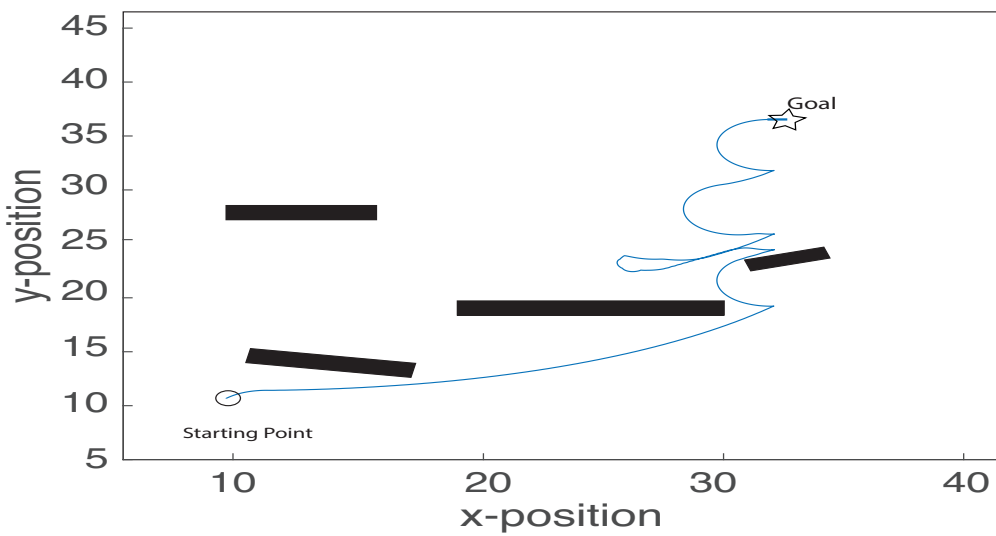


Figure 14. Path of the robot for state 4, state: Right motors are faulty (Wheel 2 and 3).

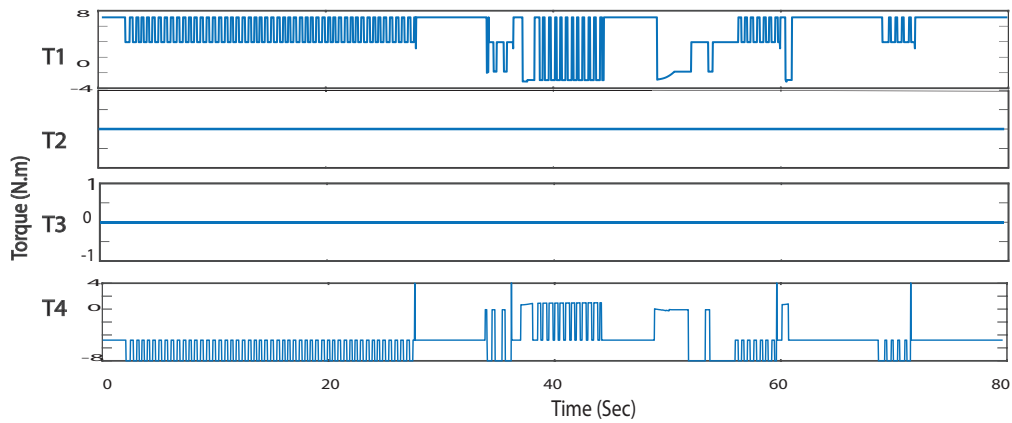


Figure 15. Simulation Results for State 4: Torque of motor 1,2,3,4.

State 5, where the rear wheels are only active, the robot was able to reach the goal smoothly. See Figures 16 and 17 that shows the path of the robot and the motors torque.

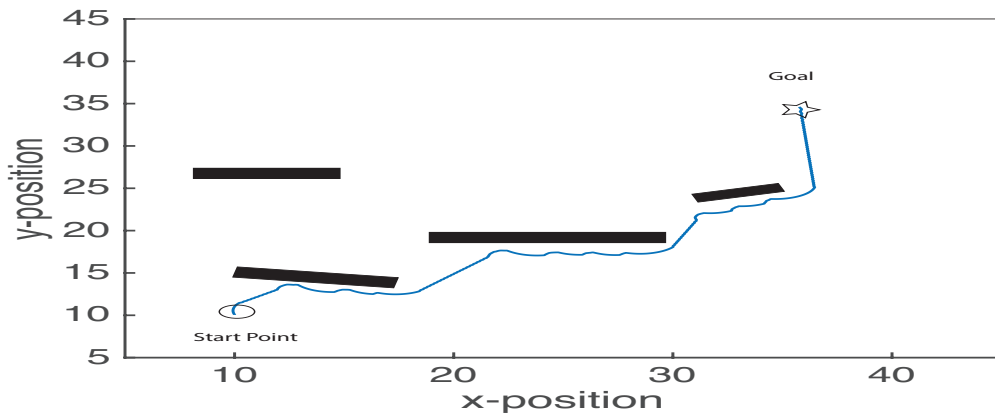


Figure 16. Path of the robot for state 5: Front wheels are faulty (Wheel 1 and 2).

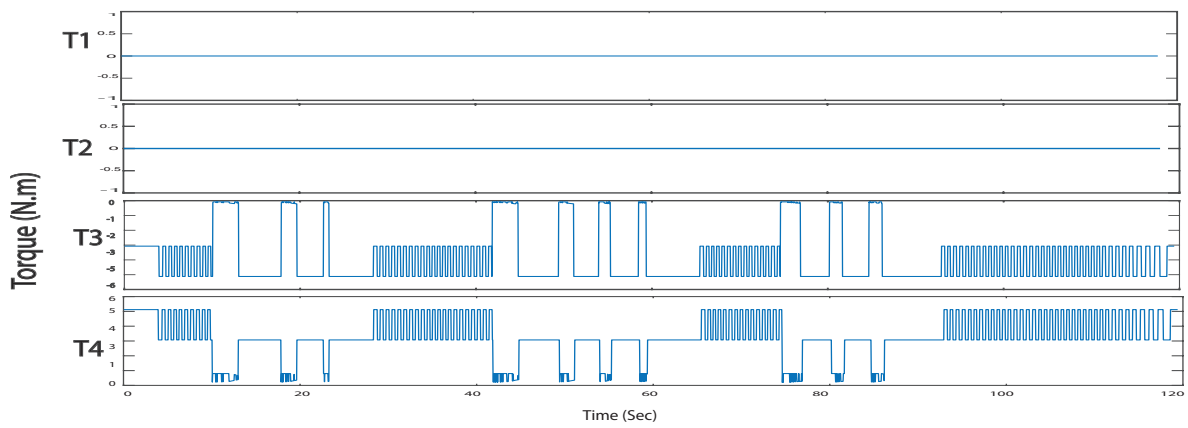


Figure 17. Simulation Results for State 5: Torque of motor 1,2,3,4.

Figure 18 shows the means error of the robot positions in the five simulated scenarios. The robot starts with an error of about 75 mod at the initial position (10,10). As the robot gets closer to the destination, the error vanishes. In all scenarios, the error goes to zero after a particular time.

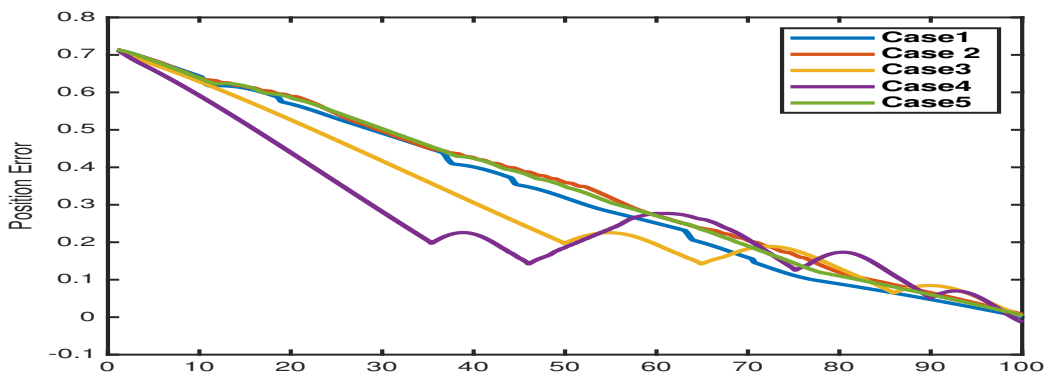


Figure 18. Position error for the five simulation scenarios.

7. Conclusions

In this paper, a fuzzy fault-tolerant control for an omnidirectional robot with four Mecanum wheels has been proposed. The motor fault is considered when the wheel can rotate freely but cannot

get the controller's command signal. Influenced by the artificial field method, fuzzy control based on the Sugeno Control principle was built. The controller consists of two main parts, Run-To-Goal and Obstacle-avoidance controller, which were built on Matlab Fuzzy Toolbox.

Several simulation scenarios (for KUKA Rob platform) were performed on the CoppeliaSim 3D simulation environment for different faults conditions. In all scenarios, the proposed algorithm could control the robot to reach its final destination with the absence and presence of an obstacle in the workspace, despite the existence of actuator faults, without crossing the workspace boundaries.

Author Contributions: A.M.A. performed the literature review, software, results analysis, Supervision, and article drafting methodology. O.A. contributed to the article writing, structuring and review of results, conceptualization. M.I. contributed to the validation, writing—original draft preparation and editing. A.G., F.M., and W.C.: writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

FTC	Fault-Tolerant Control
3D	Three-Dimensional Space
TLA	Three-letter acronym
LD	linear dichroism

References

1. Alshorman, A.; Hurmuzlu, Y. Kinematic Locomotion Modes of Particle-Based Linear Chain Mechanisms. *J. Dyn. Syst. Meas. Control* **2018**, *140*, 021010. [[CrossRef](#)]
2. Khasawneh, Q.; Jaradat, M.A.; Alshorman, A. Enhanced design of microgripper using double actuators of shape memory alloy wires. Dynamic Systems and Control Conference. *Am. Soc. Mech. Eng.* **2014**, *46186*, V001T13A004.
3. Sanaani, Y.T.; Alshorman, A.; Alshurman, K. A Novel Design of Flexure Based, Shape Memory Alloy Actuated Microgripper. ASME International Mechanical Engineering Congress and Exposition. *Am. Soc. Mech. Eng.* **2017**, *58370*, V04AT05A023.
4. Zoghoghzy, J.; Alshorman, A.; Hurmuzlu, Y. Inertially actuated baton locomotor. Dynamic Systems and Control Conference. *Am. Soc. Mech. Eng.* **2013**, *56123*, V001T10A005.
5. Jaradat, M.A.; Bani-Salim, M.; Awad, F. A Highly-Maneuverable Demining Autonomous Robot: An Over-Actuated Design. *J. Intell. Robot. Syst.* **2018**, *90*, 65–80. [[CrossRef](#)]
6. Cheng, C.; Zhu, D.; Sun, B.; Chu, Z.; Nie, J.; Zhang, S. Path planning for autonomous underwater vehicle based on artificial potential field and velocity synthesis. In Proceedings of the 2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE), Halifax, NS, Canada, 3–6 May 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 717–721.
7. Jaradat, M.A.K.; Ashour, S.M.; Matalakh, A.A.; Elayyan, M.M.; Hammadneh, A.M. Biologically inspired design of a glass climbing robot for remote services. *Int. J. Robot. Autom.* **2010**, *25*, 132. [[CrossRef](#)]
8. Mohanan, M.; Salgoankar, A. A survey of robotic motion planning in dynamic environments. *Robot. Auton. Syst.* **2018**, *100*, 171–185. [[CrossRef](#)]
9. Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous Robot Vehicles*; Springer: Berlin/Heidelberg, Germany, 1986; pp. 396–404.
10. Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. In Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, USA, 25–28 March 1985; IEEE: Piscataway, NJ, USA, 1985; Volume 2, pp. 500–505.
11. Zavrangas, P.G.; Tzafestas, S.G.; Althoefer, K. *Fuzzy Obstacle Avoidance and Navigation for Omnidirectional Mobile Robots*; European Symposium on Intelligent Techniques: Aachen, Germany, 2000; pp. 375–382.

12. Ren, L.; Wang, W.; Du, Z. A new fuzzy intelligent obstacle avoidance control strategy for wheeled mobile robot. In Proceedings of the 2012 IEEE International Conference on Mechatronics and Automation, Chengdu, China, 5–8 August 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 1732–1737.
13. Nadour, M.; Boumehraz, M.; Cherroun, L.; Puig Cayuela, V. Hybrid type-2 fuzzy logic obstacle avoidance system based on horn-schunck method. *Electroteh. Electron. Autom.* **2019**, *67*, 45–51.
14. Jahanshahi, H.; Jafarzadeh, M.; Sari, N.N.; Pham, V.T.; Huynh, V.V.; Nguyen, X.Q. Robot motion planning in an unknown environment with danger space. *Electronics* **2019**, *8*, 201. [[CrossRef](#)]
15. Karras, G.C.; Fourlas, G.K. Model Predictive Fault Tolerant Control for Omni-directional Mobile Robots. *J. Intell. Robot. Syst.* **2020**, *97*, 635–655. [[CrossRef](#)]
16. Vlantis, P.; Bechlioulis, C.P.; Karras, G.; Fourlas, G.; Kyriakopoulos, K.J. Fault tolerant control for omni-directional mobile platforms with 4 mecanum wheels. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Montreal, ON, Canada, 20–24 May 2019; IEEE: Piscataway, NJ, USA, 2016; pp. 2395–2400.
17. Siegwart, R.; Nourbakhsh, I.R.; Scaramuzza, D. *Introduction to Autonomous Mobile Robots*; MIT Press: Cambridge, MA, USA, 2011.
18. Ding, S.X. *Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms, and Tools*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2008.
19. Tzafestas, S.G. *Introduction to Mobile Robot Control*; Elsevier: Amsterdam, The Netherlands, 2013.
20. Isermann, R. *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2006.
21. Corradini, M.L.; Monteriu, A.; Orlando, G. An actuator failure tolerant control scheme for an underwater remotely operated vehicle. *IEEE Trans. Control. Syst. Technol.* **2010**, *19*, 1036–1046. [[CrossRef](#)]
22. Jiang, J.; Yu, X. Fault-tolerant control systems: A comparative study between active and passive approaches. *Annu. Rev. control* **2012**, *36*, 60–72. [[CrossRef](#)]
23. Jung, M.I.; Kim, J.H. Development of a fault-tolerant omnidirectional wheeled mobile robot using nonholonomic constraints. *Int. J. Robot. Res.* **2002**, *21*, 527–539. [[CrossRef](#)]
24. Tang, A.; Cao, Q.; Leng, C. Fault tolerant control on omnidirectional soccer robot with motor failure. In Proceedings of the 2010 8th World Congress on Intelligent Control and Automation, Jinan, China, 6–9 July 2010; IEEE: Piscataway, NJ, USA, 2010; pp. 6466–6469.
25. Ren, C.; Li, X.; Yang, X.; Ma, S. Extended state observer-based sliding mode control of an omnidirectional mobile robot with friction compensation. *IEEE Trans. Ind. Electron.* **2019**, *66*, 9480–9489. [[CrossRef](#)]
26. Song, Q.; Jiang, Z.; Han, J. Active-model-based fault tolerant control against actuator failures for mobile robot. In Proceedings of the 2006 6th World Congress on Intelligent Control and Automation, Dalian, China, 21–23 June 2006; IEEE: Piscataway, NJ, USA, 2006; Volume 1, pp. 1415–1420.
27. Mishra, S.; Sharma, M.; Mohan, S. Behavioural Fault tolerant control of an Omni directional Mobile Robot with Four mecanum Wheels. *Def. Sci. J.* **2019**, *69*, 353. [[CrossRef](#)]
28. Rotondo, D.; Puig, V.; Nejjari, F.; Romera, J. A fault-hiding approach for the switching quasi-LPV fault-tolerant control of a four-wheeled omnidirectional mobile robot. *IEEE Trans. Ind. Electron.* **2014**, *62*, 3932–3944. [[CrossRef](#)]
29. Mellah, S.; Graton, G.; El Mostafa, E.; OULADSINE, M.; PLANCHAIS, A. Detection & isolation of sensor and actuator additive faults in a 4-mecanum wheeled mobile robot (4-MWMMR). In Proceedings of the 2019 International Conference on Control, Automation and Diagnosis (ICCAD), Grenoble, France, 2–4 July 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
30. Rotondo, D.; Nejjari, F.; Puig, V. Fault tolerant control of an omnidirectional robot using a switched Takagi-Sugeno approach. In Proceedings of the 2014 IEEE International Symposium on Intelligent Control (ISIC), Juan Les Pins, France, 8–10 October 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 2183–2188.
31. Maulana, E.; Muslim, M.A.; Hendrayawan, V. Inverse kinematic implementation of four-wheels mecanum drive mobile robot using stepper motors. In Proceedings of the 2015 International Seminar on Intelligent Technology and Its Applications (ISITIA), Surabaya, Indonesia, 20–21 May 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 51–56.
32. Blyth, W.A.; Barr, D.R.; y Baena, F.R. A reduced actuation mecanum wheel platform for pipe inspection. In Proceedings of the 2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), Banff, AB, Canada, 12–15 July 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 419–424.

33. Tlale, N.; de Villiers, M. Kinematics and dynamics modelling of a mecanum wheeled mobile platform. In Proceedings of the 2008 15th International Conference on Mechatronics and Machine Vision in Practice, Auckland, New-Zealand, 2–4 December 2008; IEEE: Piscataway, NJ, USA, 2008; pp. 657–662.
34. Ratering, S.; Gini, M. Robot navigation in a known environment with unknown moving obstacles. *Auton. Robot.* **1995**, *1*, 149–165. [[CrossRef](#)]



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).