Journal of Robotics and Control (JRC)
Volume 5, Issue 1, 2024
ISSN: 2715-5072, DOI: 10.18196/jrc.v5i1.20615 

# Nonlinear Model Predictive Control-based Collision Avoidance for Mobile Robot

Omar Y. Ismael [1*], Mohammed Almaged [2], Abdulla Ibrahim Abdulla [3]
[1, 2, 3] Department of Systems and Control Engineering, Ninevah University, Mosul, Iraq
Email: [1] omar.ismael@uoninevah.edu.iq, [2] mohammed.younus@uoninevah.edu.iq, [3] abdullah.abdullah@uoninevah.edu.iq
*Corresponding Author

***Abstract*—This work proposes an efficient and safe single-layer Nonlinear Model Predictive Control (NMPC) system based on LiDAR to solve the problem of autonomous navigation in cluttered environments with previously unidentified static and dynamic obstacles of any shape. Initially, LiDAR sensor data is collected. Then, the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm, is used to cluster the (Lidar) points that belong to each obstacle together. Moreover, a Minimum Euclidean Distance (MED) between the robot and each obstacle with the aid of a safety margin is utilized to implement safety-critical obstacle avoidance rather than existing methods in the literature that depend on enclosing the obstacles with a circle or minimum bounding ellipse. After that, to impose avoidance constraints with feasibility guarantees and without compromising stability, an NMPC for set-point stabilization is taken into consideration with a design strategy based on terminal inequality and equality constraints. Consequently, numerous obstacles can be avoided at the same time efficiently and rapidly through unstructured environments with narrow corridors. Finally, a case study with an omnidirectional wheeled mobile robot (OWMR) is presented to assess the proposed NMPC formulation for set-point stabilization. Furthermore, the efficacy of the proposed system is tested by experiments in simulated scenarios using a robot simulator named CoppeliaSim in combination with MATLAB which utilizes the CasADi Toolbox, and Statistics and Machine Learning Toolbox. Two simulation scenarios are considered to show the performance of the proposed framework. The first scenario considers only static obstacles while the second scenario is more challenging and contains static and dynamic obstacles. In both scenarios, the OWMR successfully reached the target pose (1.5m, 1.5m, 0°) with a small deviation. Four performance indices are utilized to evaluate the set-point stabilization performance of the proposed control framework including the steady-state error in the posture vector which is less than 0.02 meters for position and 0.012 for orientation, and the integral of norm squared actual control inputs which is 19.96 and 21.74 for the first and second scenarios respectively. The proposed control framework shows a positive performance in a narrow-cluttered environment with unknown obstacles.**

***Keywords*—Nonlinear Model Predictive Control (NMPC); DBSCAN; Set-point Stabilization; Obstacle Avoidance; Mobile robot; Real-time Navigation; CasADi; CoppeliaSim.***

## I. INTRODUCTION

Yet, as autonomous mobile robots spread, new technological problems appear, shedding more light on the difficulties in developing secure and a dependable intelligent motion system that can cooperate with people and complicated environments that contain dynamic and static obstacles. It is difficult to maintain autonomy in such a dynamic, and unstructured environment. The following presents the most challenging areas: i) reliable obstacle detection and forecasting in an unstructured environment, ii) examination of obstacles' uncertainty and parametric representation, and iii) algorithm of motion planning for dynamic obstacle trajectories in real-time.

In addition to the unique tasks that each mobile robot is intended for [1]-[7], dynamic control problems and path planning are critical when autonomous navigation is being considered. Planning and control problems are frequently dealt separately in the literature using multi-layer schemes [8]-[13]. If the connection between the layers is not handled properly in certain circumstances, the total output might show performance degradation. Control systems in particular frequently assume that the intended reference is attainable, though this hypothesis may not be accurate if the reference is specified despite the system's inherent dynamics limitations and operational constraints. Thus, it is crucial to research methods that take control and planning problems into account in a single, colligated scheme.

Among the studies that address these problems at single-layer, those that focus on optimum control are particularly intriguing since they allow for the definition of path planning by utilizing the design of the optimization problem through its cost function and constraints [14][15]. From the standpoint of the control algorithm, Model Predictive Control (MPC) techniques in particular can provide convergence guarantees and online stability while solving the path planning problem. [16]. MPC solves an optimization problem to determine the optimal action of control by using the Multi-Input Multi-Output (MIMO) model of the mobile robot to forecast its future behavior. It is frequently difficult to control a MIMO mobile robot system using conventional controllers like PID because of the interactions between the interdependent inputs and outputs. Nevertheless, MPC can account for input-output interactions and concurrently regulate all of the outputs. MPC offers feed-forward control-like preview capabilities. Advance knowledge of set point changes enables the MPC to adjust and perform better in response to those changes. However, the complexity of MPC's algorithm, which requires more time than other controllers, is a drawback. However, the advancement in processing power of recently invented microprocessors solves this problem. For the aforementioned benefits, MPC has become more popular in the control community.

Different versions of MPC, such as linear MPC and NMPC, have been employed to address the aforementioned

objectives of control for mobile robots. In Ref. [17]-[19] provide research that employed linear MPC; in these investigations, MPC has only been utilized to accomplish the objective of path or trajectory tracking. The nonlinear model of motion is employed by NMPC, which has been utilized for stabilizing problems [20]-[27]. NMPC has several advantages such as, it does not utilize iterative algorithms; it may acquire the control action using the same techniques as linear MPCs; it can be used for robot nonlinear models. Although stability of a finite horizon NMPC is not trivially guaranteed [28][29], it has been demonstrated that stability may be ensured by employing a terminal state equality constraint [30]-[32].

Both linear MPC and NMPC can explicitly handle the constrained control problems. Constraints are crucial since going against them might have unfavorable effects. The majority of research in the literature suggests single-layer algorithms while taking into account that NMPC creates extra constraints for obstacle avoidance to solve the problem [33][34]. Over a predefined prediction horizon, NMPC determines a future control sequence while reducing an objective function where a set of system state and control action constraints are achieved. Analytical techniques can be utilized to include obstacle avoidance needs as constraints in the optimization problem, based on the concept of combining control and planning strategies. Several studies in the literature have examined stabilizing NMPC schemes as a cost for avoidance in the context of penalty functions [35], and artificial potential fields [36]. However, these studies only consider static obstacles cases.

Mobile robots must be able to recognize surrounding obstacles quickly and effectively, and then include them as constraints inside the NMPC, to perform safety-critical navigation in dynamic obstacle situations. These can be achieved by the developments in sensor technology. Mobile robots, for instance, may look ahead a specific distance using LIDAR, Radar, and cameras, delivering information about the terrain and other traffic data [37]. Based on visual data from cameras, several approaches for object detection and tracking are utilized [38]-[42]. Nevertheless, in poor lighting, adverse weather, and cluttered environments this strategy might not work effectively. A different approach uses a LiDAR sensor for object detection. This approach performs well in poor weather, dim illumination, and crowded surroundings.

Many techniques utilize the LiDAR sensor for object detection in the literature. The most effective techniques use point cloud data. In Ref. [43] developed a system for dynamic environment perception-based point cloud data. In this method the dynamic obstacles are enclosed with minimum bounding ellipses, and a stable dynamic obstacle avoidance is achieved. However, in cluttered environments, when there is a short distance between two or more obstacles, the ellipsoids or other shapes can overlap with each other and then, prevent the robot from navigating between these obstacles especially, when the path is narrow. This may result in the loss of recursive feasibility, in the way of avoiding obstacles that demand a large detour from the initial path.

In this work, obstacles are clustered by the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm [44] after collecting the LiDAR sensor data. DBSCAN works effectively with clusters of any shape. DBSCAN is robust to outliers and has a concept of noise. It is not necessary to define the number of clusters in advance when using DBSCAN. On the other hand, choosing the DBSCAN algorithm parameters can be challenging. Nevertheless, this can be solved if the collected LiDAR sensor data is well understood. Another problem with using the DBSCAN algorithm is that a boundary point can belong to either cluster if it can be reached from more than one cluster. This problem is not important in the proposed work since it utilizes the minimum Euclidean distance between the robot and each obstacle is calculated. The proposed algorithm performs well in a narrow-cluttered environment with unknown static and dynamic obstacles.

Numerous techniques, including artificial potential fields [45], collision-free flight corridor [43][46], DWA (Dynamic-Window Approach) [47], gradient maps [48], social forces [49], and pre-computed motion primitives library [50][51], can be used to avoid collisions in static and dynamic environments. These techniques, however, are ineffective in contexts with more complicated or fast-moving obstacles. The proposed strategy is based on MPC, which has been increasingly popular recently [52-][54]. In Ref. [43] presented a planning strategy in dynamic, unstructured environments based on Model Predictive Contouring Control (MPCC). Nevertheless, this algorithm only effectively avoids pedestrians, making it challenging to successfully avoid more complicated obstacles. To assure safety, [55] suggested a model predictive control architecture, and [56] examined its viability and safety by employing discrete time Dynamic Control Barrier Function (D-CBF) limitations in a receding horizon manner. This algorithm, however, struggles in a narrow-cluttered environment.

In this paper, dynamic obstacles in a narrow environment can be handled. This work proposes an NMPC algorithm with obstacle avoidance capabilities for a set-point stabilization motion system. The DBSCAN algorithm is adopted for clustering the dynamic and static obstacles after collecting the LiDAR sensor information. The environmental obstacles are induced as additional constraints, which are represented in this work as having minimum distance forms that are effective in situations with fast-moving or more sophisticated shape obstacles. Numerical findings are provided taking into account the set-point stabilization methodology used for an OWMR with holonomic constraints to support the suggested control strategy. A simulation is executed using the CoppeliaSim robot simulator in conjunction with MATLAB R2023a, which utilizes the CasADi Toolbox [57] with the Interior Point OPTimizer (IPOPT) solver [58] and Statistics and Machine Learning Toolbox, to evaluate the performance of the proposed method.

This work's contributions may be summed up as follows:

- New set-point stabilization control framework to establish collision-free trajectory in static and dynamic environments with previously unknown obstacles of any shapes, by inducing these obstacles as additional

inequality constraints that can be easily included inside the NMPC. This combines the obstacles avoidance algorithm with the set-point stabilization NMPC, which saves time and effort.

- It proposes a safe method for representing obstacles as a single point based on minimum Euclidean distance rather than existing methods in the literature that depend on enclosing the obstacles with a circle or minimum bounding ellipse. This helps the proposed method to handle numerous obstacles at the same time rapidly and efficiently in unstructured environments with narrow corridors.
- The effectiveness and real-time performance of the obstacle avoidance algorithm are tested using the CoppeliaSim robot simulator in experiments similar to real-world scenarios that contain multi-static and dynamic obstacles of different shapes, and corridors.

The rest of this paper is structured as follows: The kinematic model of the OWMR is described in Sect. 2. The NMPC framework for obstacle avoidance is presented in Sect. 3, numerical results are provided in Sect. 4 to support the suggested control strategy, and the study is concluded in Sect. 5.

## II. KINEMATIC MODEL OF OWMR

The body frame ($X_\mathrm{b}$, $Y_\mathrm{b}$) and the global frame ($X_\mathrm{g}$, $Y_\mathrm{g}$) are the coordinate frames utilized in the modeling of the OWMR as illustrated in Fig. 1. The body frame is attached to the origin of the moving robot while the global frame is assigned to the fixed ground. $\theta$ is the angle that indicates the robot's orientation in the world frame. This work assumes that the robot travels on a flat surface and is able to travel wherever in the environment outside of the region where obstacles are present. Furthermore, is assumes that the robot moves with no slipping. The wheel slipping can be avoided when the rollers come into contact with most materials, they have a high sliding friction coefficient. This is can be ensured by making the rollers from rubberlike material, and having a large robot's size. In addition, the proposed NMPC in section 3 reduces robot speed in order to prevent slippages. All of the robot's dynamic parameters and geometric are known all the time.

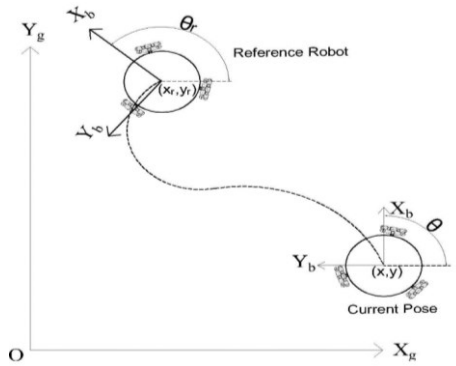


Fig. 1. OWMR frames and variables definition

The position and orientation (pose) of the OWMR in the global coordinates can be described by the following state vector (1).

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{1}$$

where $x$ and $y$ are the robot position, $\theta$ is the angle between the $x$-axes of the robot and global frames. In terms of robot body velocities which is characterized by $\mathbf{u} = [v_x \quad v_y \quad \omega]^T \in \mathbb{R}^3$; $\mathbf{u}$ is the control input vector, whereas $v_x, v_y$ and $\omega$ symbolize robot translational and rotational velocities respectively, and can be expressed in the global coordinates as follows OWMR kinematic model [59][60]:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = R^T(\theta)\,\mathbf{u} \tag{2}$$

where $R(\theta)$ denotes the orthonormal rotation matrix that transforms between the robot's coordinate system and the global coordinate system and can be written as:

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

By substituting (2) in (1):

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix}$$

The system (3) is controllable as it has its accessibility rank condition globally satisfied, and is in the control-affine form (4).

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} v_x + \begin{bmatrix} -\sin\theta \\ \cos\theta \\ 0 \end{bmatrix} v_y + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega \tag{4}$$

The three Omni-wheels are placed relative to the robot frame at angles $\alpha = [\alpha_1 \quad \alpha_2 \quad \alpha_3]^T \in \mathbb{R}^3$. All wheels are non-steerable and perpendicular to the robot circumference. Taking the wheel velocities into consideration and with respect to the robot body coordinates, the lower level kinematic model can be defined as:

$$\dot{\boldsymbol{\phi}} = \frac{1}{r} \begin{bmatrix} \sin\alpha_1 & -\cos\alpha_1 & L_\mathrm{g} \\ \sin\alpha_2 & -\cos\alpha_2 & L_\mathrm{g} \\ \sin\alpha_3 & -\cos\alpha_3 & L_\mathrm{g} \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{5}$$

By placing the Omni-wheels at $\alpha_1 = \pi/3$, $\alpha_2 = \pi$, and $\alpha_3 = -\pi/3$, the lower-level kinematic model can be written as:

$$\dot{\boldsymbol{\phi}} = \frac{1}{r} \begin{bmatrix} \sqrt{3}/2 & -1/2 & L_\mathrm{g} \\ 0 & 1 & L_\mathrm{g} \\ -\sqrt{3}/2 & -1/2 & L_\mathrm{g} \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix}$$

where the $\dot{\boldsymbol{\phi}} = [\dot{\phi}_1 \quad \dot{\phi}_2 \quad \dot{\phi}_3]^T \in \mathbb{R}^3$ represents the vector of wheels angular velocities, and $r$ denotes the radius of each wheel of the robot. Every wheel is in the OWMR is identical and connected to an electrical motor, and spaced equally by ($L_\mathrm{g}$) to the center of mass of the OWMR ($R$). The maximum wheel velocity depends on the specifications of this electrical motor. It is constrained by $\dot{\phi}_{max}$, namely $\forall i : \dot{\phi}_i \leq \dot{\phi}_{max}$, where the subscript ($i$ = 1, 2, 3) represents the $i^{th}$ wheel velocity since the motor's voltage and current are magnitudes

restricted by the resistance of the coils, and the counter electromotive force.

As shown in Fig. 1, a reference robot is defined to demonstrate the objective of the control algorithm which is the stabilization of an OWMR to a permissible equilibrium. This reference robot is subjected to the same constraints as system (1), and possesses a reference state vector $\mathbf{x_r} = [x_r \quad y_r \quad \theta_r]^T \in \mathbb{R}^3$ and a reference control vector $\mathbf{u_r} = [v_{xr} \quad v_{yr} \quad \omega_r]^T \in \mathbb{R}^3$. Therefore, the kinematic motion model can be represented as follow:

$$\dot{\mathbf{x}}_{\boldsymbol{r}} = \begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} \cos\theta_r & -\sin\theta_r & 0 \\ \sin\theta_r & \cos\theta_r & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_{xr} \\ v_{yr} \\ \omega_r \end{bmatrix} \tag{6}$$

The reference control vector $\mathbf{u_r}$ has zero values for both angular and linear velocities, and the reference state vector $\mathbf{x_r}$ has a constant value corresponding to the desired target pose for the set-point stabilization problem.

The primary goal of the control algorithm at this point can be specified as simultaneous robot stabilization while providing safe navigation. The controller ought to also take into consideration the map boundaries, previously unknown obstacles, and the robot geometry. Furthermore, to design a practical controller, the robots' inputs saturation margins must be considered. The designed NMPC architecture shown in the following section, addresses the aforementioned difficulties and problems.

## III. NONLINEAR MODEL PREDICTIVE CONTROL

NMPC is considered to be one of the most reliable optimal control techniques and it is utilized in this work due to its ability to easily deal with the system constraints and take future prediction into the controller design in regard to nonlinear system defined through the following differential equation:

$$\dot{\mathbf{x}}\,(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \qquad \mathbf{x}(0) = \mathbf{x}_0 \tag{7}$$

where $f : \mathbb{R}^3 \times \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is the nonlinear mapping created by the robot model (7).

The objective of the controller is to determine an admissible control input $\mathbf{u}^*(t)$ that will lead system (7) to drive toward the equilibrium point described by:

$$\begin{aligned} \mathbf{x}_e(t) &= \mathbf{x}_r(t) - \mathbf{x}(t) = 0 \\ \mathbf{u}_e(t) &= \mathbf{u}_r(t) - \mathbf{u}(t) = 0 \end{aligned} \tag{8}$$

The aim of the control technique is to produce a minimum weighted cost function ($J$) over a prediction horizon ($T$) as described in (9) [61]:

$$J(t, \mathbf{x}_e(t), \mathbf{u}_e(t)) = P(\mathbf{x}_e(t+T)) + \int_t^{t+T} \ell(\tau, \mathbf{x}_e(\tau), \mathbf{u}_e(\tau))\, d\tau \tag{9}$$

Here, $\ell(\tau, \mathbf{x}_e(\tau), \mathbf{u}_e(\tau))$ is the running cost function that is integrated over $T$ that is obtained from (10).

$$\ell(\tau, \mathbf{x}_e(\tau), \mathbf{u}_e(\tau)) = \mathbf{x}_e{}^T(\tau) Q \mathbf{x}_e(\tau) + \mathbf{u}_e{}^T(\tau) R \mathbf{u}_e(\tau) \tag{10}$$

The terminal state penalty $P(\mathbf{x}_e(t+T))$, assessed at the last step of the optimization horizon is presented as:

$$P(\mathbf{x}_e(t+T)) = \frac{1}{2}\, \mathbf{x}_e{}^T(\tau) F \mathbf{x}_e(\tau) \tag{11}$$

The $Q$, $R$ and $F$ represent the positive definite symmetric weight matrices while and $\mathbf{x}_e$ and $\mathbf{u}_e$ define the robot's state and control vector errors respectively. At time ($t$) to avoid obstacles in the environment, NMPC optimization problem can be formed as following:

$$\begin{aligned} &\min_{\mathbf{u}^*} \; J(t, \mathbf{x}_e(t), \mathbf{u}_e(t)) \\ \text{Subject to:} \quad &\dot{\mathbf{x}}\,(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \\ &\mathbf{x}(t) \in X, \;\; (\tau \in [t, t+T]) \\ &\mathbf{u}(t) \in U, \;\; (\tau \in [t, t+T]) \end{aligned} \tag{12}$$

At which the set $X \in \mathbb{R}^3$ and the set $U \in \mathbb{R}^3$ identify the allowable state and control sets as determined by the upcoming sets of constraints. Firstly, the box constraints for the map boundaries and the OWMR input saturation limits that are provided by:

$$\begin{aligned} x_{min} &\le x \le x_{max} \\ y_{min} &\le y \le y_{max} \\ v_{x\,min} &\le v_x \le v_{x\,max} \\ v_{y\,min} &\le v_y \le v_{y\,max} \\ \omega_{min} &\le \omega \le \omega_{max} \end{aligned} \tag{13}$$

The map margins are indicated by the sets ($x_{min}$, $x_{max}$) in the $x$-axis and ($y_{min}$, $y_{max}$) in the $y$-axis, and the robot input saturation limit described by ($v_{x\,min}$, $v_{x\,max}$), ($v_{y\,min}$, $v_{y\,max}$) for translational velocities, and ($\omega_{min}$, $\omega_{max}$) for rotational velocities.

Secondly, in execution time, the obstacles are described as points rather than shapes by using LiDAR sensor and DBSCAN algorithm which make it suitable for narrow paths. Based on the system's global location and the environment map, a detection algorithm is developed to detect obstacles within a range of $M$ meters, since the obstacles are presumed to have been unknown to the controller previously. As a result, one can only avoid obstacles while they are within the sensor's range. In order to support the robot to have a safe navigation in the map while avoiding obstacles, it is required to consider several environmental factors such as the robot radius, the distance between the robot and the obstacles, …. etc. The following environmental constraint is taken into account to avoid dynamic and static obstacles:

$$\forall k : \sqrt{((x - xobs_k)^2 + (y - yobs_k)^2} \ge (R_b + \beta) \tag{14}$$

where $k = 1, 2, \ldots n$, $n$ is a predetermined number of previously undefined obstacles that are clustered by the DBSCAN algorithm after collecting the LiDAR sensor information. $xobs_k$, $yobs_k$ are the coordinates of the nearest point in the obstacle surface to the robot, $R_b$ is the radius of the robot that indicate its influence on the map, and $\beta$ is the safety margin that can be added to enhance the safe navigation of the motion system when inside the non-obstructed space (i.e. bigger $\beta$ means a safer navigation due to overcome the distance reading error).

The proposed framework assumes that certain online information about obstacles is anticipated using

measurement or estimation for control reasons. This raises questions about whether it is feasible to get reliable information in real-time, particularly in highly dynamic or quickly changing environments, which might result in delayed or erroneous robot response. However, slowing down the robot velocity, choosing the appropriate $\beta$, and utilizing omni-LiDAR sensor can make the proposed method more reliable. In the other hand, it is important to note that no prior information of the number, size, or geographic distribution of obstacles is necessary.

Finding the most suitable terminal penalty and constraints is made possible by the stability theory provided in [22][62] as following:

- A continuous cost function is assumed with $\ell(0,0) = 0$ and $\ell(\mathbf{x}_e, \mathbf{u}_e) > 0$.
- Consider that the reference control signals are bounded, i.e., $[v_{xr} \quad v_{yr} \quad \omega_r]^T < [v_{xrmax} \quad v_{yrmax} \quad \omega_{rmax}]^T$, and at time $t = 0$, the open-loop optimization problem retains a definite solution.
- $P(\cdot)$ is presumed to be differential and continuous function, and fulfilling $P(\mathbf{0}) = 0$, and $P(\mathbf{x}_e) > 0$ for all $\mathbf{x}_e \neq 0$.
- A terminal-state controller $\mathbf{u}_e^{\mathrm{L}}$ exists such that the following condition is satisfied:

$$\dot{P}(\mathbf{x}_e) + \ell(\mathbf{x}_e, \mathbf{u}_e) \leq 0, \forall\, \mathbf{x}_e \in \Omega \tag{15}$$

where $\Omega$ represents the terminal-state region. Then, the closed-loop system is guaranteed to be asymptotically stable using the NMPC approach previously stated [22][62].

The computation time of the constrained nonlinear optimization remains an issue for real-time implementation, even if the stability of the NMPC is ensured by Ω constraint and $P$. Even if there is an optimization solution, it might not be feasible to locate it in the restricted control time. As in [22], the optimal control profile is not necessary for the aforementioned stability analysis. Stability can be obtained from any feasible profile of control. It implies that by locating feasible solutions in the optimization problem, a stable closed control can be attained. Thus, optimization is accomplished if the optimal solution can be discovered within the restricted number of optimization steps. If not, a feasible solution is discovered and the optimization algorithm halts at the restricted optimization steps to ensure real-time control. At last, one attains a suboptimal controller.

The calculation is a major problem to employing NMPC in real-time systems, in addition to the stability issue. To solve this problem, in this work, the NMPC creates a control profile for every optimization step. The robot receives just the first control signal from the control profile; all other control signals are ignored. The controller needs to solve the constrained nonlinear optimization problem once again in the next step. But for the present optimal solution, the preceding control profile serves as positive "hot start" or initial solution. Using "hot start" in the optimization computation reduce the computation time. In addition, for real-time applications, the computation can be reduced when the predictive control horizon is chosen to be as short as possible. Finally, variations in cost function weight parameters may lead to variations in control performance.

## IV. SIMULATION AND RESULTS

The simulation for the OWMR dynamics and the environment is implemented inside the CoppeliaSim robotic simulator. CoppeliaSim is a very powerful robotic simulator that supports rigid and soft bodies dynamics simulation. Moreover, it contains numerus built-in robot models, tools, sensors, and actuators that can be utilized to build a virtual environment with ability for real-time interaction. CoppeliaSim can be programmed with various programming languages such as Lua, MATLAB, C++, and Python which makes it ideal choice for robot simulations. Before implementing the proposed control framework on actual hardware, this work utilizes the CoppeliaSim to verify it in a virtual environment. CoppeliaSim speeds up the design and implementation process by making it easier to develop and test robotic systems.

In this simulation setup, it is assumed that OWMR knows its pose $(x, y, \theta)$ at each simulation step. Additionally, it is equipped with a laser rangefinder with a range of 3m and 360° field of view to detect the obstacles around the OWMR. However, the assumption of the OWMR knowing its pose at each step might not take the error and uncertainty in robot pose practical implementation. A solution for this can be by utilizing a vision-based motion capture system to provide the robot with position and orientation input to guarantee precise localization [63].

MATLAB is used for laser rangefinder data processing and NMPC implementation. The connection between CoppeliaSim and MATLAB is done using ZeroMQ remote APIs. It provides cross-platform communication, permits communication between CoppeliaSim and an external MATLAB program (that is, an application operating on a separate system or in a different process), and allows the same API function calls as those made from within a CoppeliaSim script. The integration eliminates the requirement for any setup by making the CoppeliaSim API directly accessible to the remote client (in MATLAB). ZeroMQ's robust messaging library and great performance are the foundation of communication and messaging API. MATLAB receives raw data from the laser range finder and uses the DBSCAN algorithm inside the Statistics and Machine Learning Toolbox to cluster the data points that belong to each obstacle together. The output of the DBSCAN algorithm are set of points where each one represents an obstacle in the scene. The minimum distance between each set of points and the center of the OWMR is then calculated. Furthermore, the CasADi toolbox inside MATLAB is used to perform both the system states integration, and optimal control computation. The NMPC is implemented in the CasADi toolbox, where both the system model (7), and the optimal control problem are defined. By using the multiple shooting method and IPOPT solver for nonlinear programming problems, the optimum control problem is resolved, and states integration is accomplished. The convergence criterion of IPOPT is kept at $10^{-8}$ and the maximum number of iterations was set to 2000. More information about CasADi toolbox for MATLAB is available

in [52]. Runge-Kutta 4th (RK4) order method is employed for states integration. On a core i7 personal computer with 16 GB RAM and a 2.59 GHz CPU, all simulations were run.

The DBSCAN algorithm needs two parameters MinPoints and Epsilon. MinPoints specifies the bare minimum of points required to create a dense region. Epsilon establishes the minimum distance between two points for them to be classified as a cluster. In this work, MinPoints is selected as 3, and Epsilon is chosen as 0.1. The controller saturation limitations have been chosen from [59] and the map margins have been adopted from [17], as follows:

$$\begin{gathered} -0.5 \leq v_x \leq 0.5\ (m/s) \\ -0.5 \leq v_y \leq 0.5\ (m/s) \\ -\pi/4 \leq \omega \leq \pi/4\ (rad/s) \\ -2 \leq x \leq 2\ (m) \\ -2 \leq y \leq 2\ (m) \end{gathered}$$

From [59][63], the radius of the robot in (14) is chosen as $R_b = 0.1\ (m)$, and robot wheels radii in (5) is the are selected as $r = 0.035\ (m)$. The safety margin is selected same as half of the robot radius as $\beta = 0.05\ (m)$.

One parameter in the NMPC is the time step $\Delta T$. The rejection of the unknown disturbance often becomes better as $\Delta T$ drops. On the other hand, the computing effort rises sharply as $\Delta T$ gets smaller. Therefore, the best option strikes a compromise between computational effort and performance. Faster calculations are encouraged by small m since it requires fewer variables to be computed in the QP solution at each control interval. total number of horizon steps N, is another crucial factor in NMPC. The QP solution time and controller memory increase as the N increases. Nevertheless, using a shorter prediction horizon also increases the controller's aggressiveness. N must vary inversely with $\Delta T$ if one decides to retain the prediction horizon duration (the product $\mathrm{N} \times \Delta T$) constant.

It is advised to establish N early in the controller design and to maintain it at that value when adjusting other NMPC parameters, including the weights of the cost functions $(Q, R, F)$ to compensate for increments in $\mathbf{x}_e$ and $\mathbf{u}_e$ and drive them to zero with the desired performance. Rather, N should be set so that the controller is internally stable and can detect breaches of constraints in time to take remedial action. The prediction horizon parameters are determined as follows: the total number of horizon steps $\mathrm{N} = 5$ (steps) and the time step $\Delta T = 0.2$ (seconds), resulting in a prediction horizon length of $\mathrm{T} = 1$ (second). It is decided to run the simulation for 10 (seconds) in total. The weight matrices appear in (10 & 11) are selected as:

$$Q = F = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}, \text{and } R = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix}$$

To show the performance of the proposed collision avoidance scheme, two simulation scenarios are considered. In both scenarios, the robot is required to go to the target pose of $\mathbf{x}_r = (1.5m, 1.5m, 0°)$, and the map margins are highlighted with a black edged box.

The first scenario is shown in Fig. 2 where there are only static obstacles in the scene, the robot starts from the initial configuration specified by $\mathbf{x}_0 = (-0.6m, -2m, 0°)$. As can be seen from Fig. 2, there is a narrow passage of 0.475m width. The OWMR successfully passes through the passage and then encounters a cuboid-shaped obstacle when exiting the passage which it bypasses and reaches the goal pose.

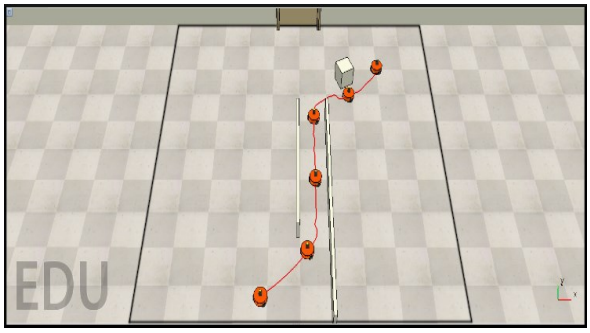


Fig. 2. OWMR performed trajectory for the first scenario

Fig. 3 illustrates the state vector $\mathbf{x}$ components of the OWMR for static obstacles avoidance scenario under the proposed controller. Along the simulation time, the proposed controller exhibits a smooth transition of the robot's states. Moreover, the states demonstrate a rapid convergence to their reference values. Fig. 4 shows the control actions (linear and angular) applied to the OWMR for not moving obstacles avoidance case. As can be deduced from this figure, the proposed controller shows a non-aggressive control action, and did not exceed their saturation limits.

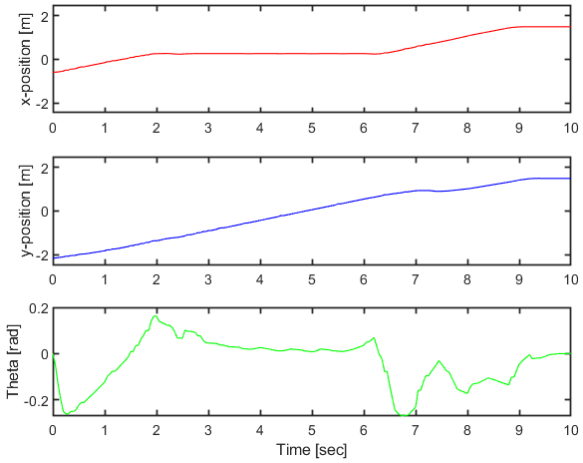


Fig. 3. OWMR state vector $\mathbf{x}$ components for the first scenario

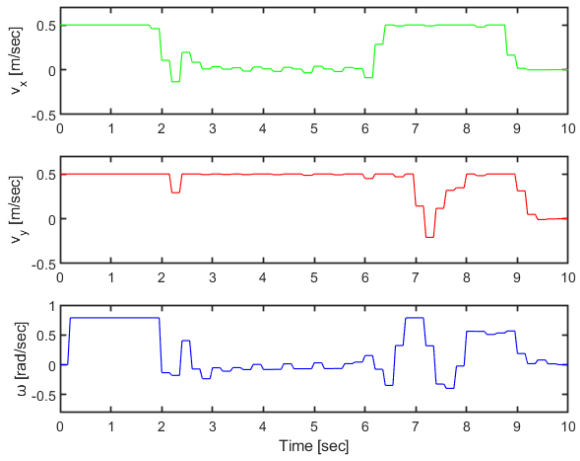


Fig. 4. OWMR controls vector $\mathbf{u}$ components for the first scenario

The second scenario is more challenging and includes moving obstacles in addition to the static ones as presented in Fig. 5. In this scenario, the robot starts from the initial configuration specified by $\mathbf{x}_0 = (-0.6m, -1.7m, 0°)$, and there are three moving obstacles which are colored in cyan, red, and blue. There are two static obstacles colored in pink close to the target position, green disk. As shown in Fig. 5, the OWMR encounters the cyan obstacle at first and avoids it successfully. After that, it faces the red cuboid and bypasses it, and then avoids the blue obstacle. Finally, it passes between the static obstacles and reaches the target pose. To ensure dynamic obstacle avoidance, the following condition is imposed: the speed of the obstacle should be less than the speed of the robot.

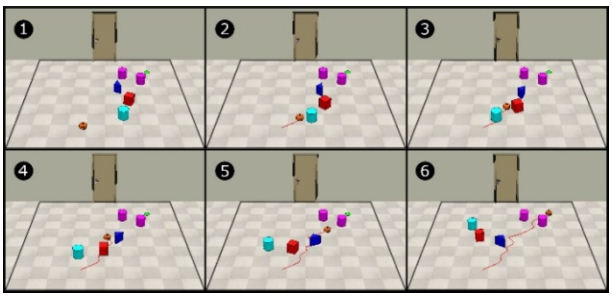


Fig. 5. OWMR performed trajectory for the second scenario.

Fig. 6 shows the OWMR state vector $[x \quad y \quad \theta]^T$ components under the proposed controller for moving obstacles avoidance case. The proposed controller shows a smooth change in the robot's states throughout the simulation. Additionally, the states demonstrate a quick convergence to their reference values. Fig. 7 depicts the OWMR's control action for the dynamic obstacles avoidance scenario. As can be seen from the figure, the proposed controller did not exceed its saturation limitations and exhibited a smooth control action.

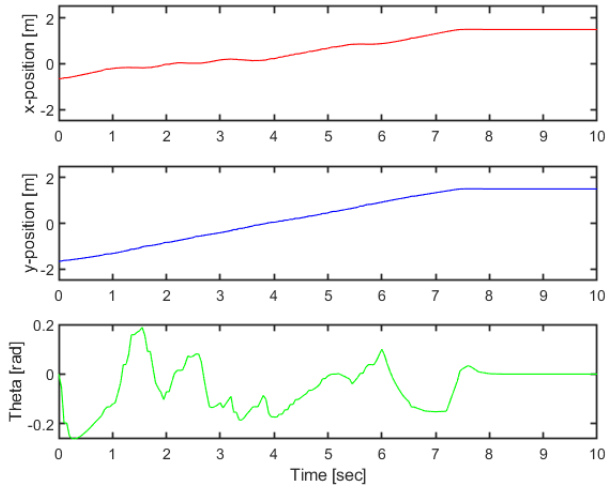


Fig. 6. OWMR state vector **x** components for the second scenario

The performance indices of the set-point stabilization results under the proposed controller are shown in Table I. The steady-state error in the posture vector for the OWMR ($x_e = x_r - x$, $y_e = y_r - y$, and $\theta_e = \theta_r - \theta$) is the first performance metric. The integral of norm squared actual control inputs $\gamma = \sum_{i=1}^{\alpha} \left( \|\mathbf{u}_i\|^2 \Delta T \right)$ is the second performance index to evaluate the control energy, where $\alpha$ is the number of updating steps for the controller. The average computing capacity $C$ is the third performance metric.

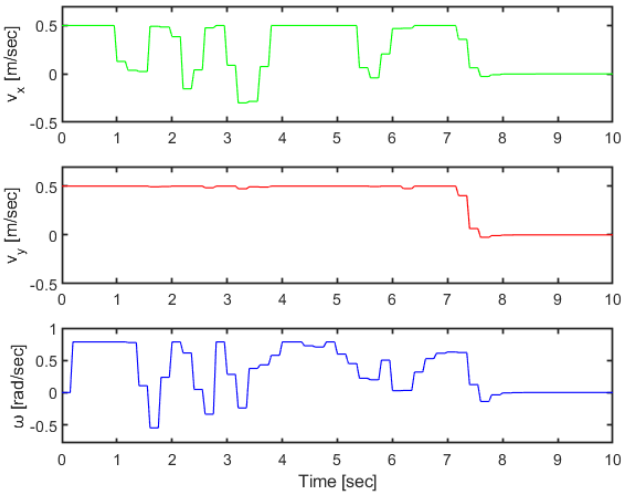


Fig. 7. OWMR controls vector **u** components for the second scenario

As seen in Table I, the robot is demonstrated to have reached the target destination in both scenarios with error of less than 0.02 (meters) for position and less than 1 (degree) for orientation. The integral of norm squared actual control inputs is 19.96 and 21.74 for the first and second scenarios respectively. The dynamic obstacle avoidance scenario needs more control effort compared with the static one because the robot needs to change its orientation more to be able to handle the moving obstacles. It has been noted that the average computing capacity needed for the proposed controller is 0.5 (seconds) for the first scenario and 0.4 (seconds) for the second scenario for each time step. In fact, the computing capacity need would rise with the number of obstacles.

TABLE I. SET-POINT STABILIZATION PERFORMANCE INDECIS

| Scenario | Performance Indices | | | | |
|---|---|---|---|---|---|
| | $x_e(m)$ | $y_e(m)$ | $\theta_e(rad)$ | $C(sec)$ | $\gamma$ |
| 1. Static obstacles | 0.013 | 0.018 | 0.009 | 0.5 | 19.96 |
| 2. Dynamic obstacles | 0.017 | 0.019 | 0.012 | 0.4 | 21.74 |

## V. CONCLUSION

In this paper, an NMPC framework was used to stabilize OWMR to a specific target while avoiding obstacles in a cluttered environment. The DBSCAN algorithm was implemented for clustering the dynamic and static obstacles after collecting the information from the LiDAR sensor. The environmental obstacles were considered as further constraints, which are denoted in this work as having minimum distance forms. A simulation was performed using MATLAB that utilizes the CasADi Toolbox with IPOPT solver, and Statistics and Machine Learning Toolbox, in conjunction with the CoppeliaSim robot simulator. Two scenarios were considered based on whether the obstacle was stationary or moving. To evaluate the performance of the proposed control framework for the set-point stabilization, four performance metrics were applied counting the steady state error in the pose vector, and the integral of norm-squared actual control inputs. During numerical simulations

of OWMR stabilization in environments with static and dynamic obstacles, the proposed controller algorithm has demonstrated acceptable performance in real-time according to the aforementioned metrics. The proposed controller's scalability was restricted to a large number of obstacles, even though it demonstrated higher performance and satisfied real-time requirements. Furthermore, regardless of whether the simulation results consistently demonstrate that the OWMR converges to its references, it was still necessary to look at the stability analysis of the chosen approach in this case.

The future work of this research includes the practical implementation of the proposed control framework on a real OWMR. The following recommendations can be taken into account for the future practical implantation:

- The assumption of the OWMR knowing its pose at each step in simulation setup might not take the error and uncertainty in robot pose practical implementation. A solution for this can be by utilizing a vision-based motion capture system to provide the robot with position and orientation input to guarantee precise localization [64].
- To reduce the dynamic effects that are not taken into account by the kinematic model, the OWMR may need to operate at slower speeds than in the simulations. Observer-based controllers, as those described by [65]-[67], can be integrated with the proposed controller by estimating such environment characteristics to quantify model disturbances and uncertainties.

## REFERENCES


[1] G. Li *et al*., "Hybrid Maps Enhanced Localization System for Mobile Manipulator in Harsh Manufacturing Workshop," *IEEE Access*, vol. 8, pp. 10782–10795, 2020, doi: 10.1109/access.2020.2965300.

[2] P. Tokekar, J. Vander Hook, D. Mulla, and V. Isler, "Sensor planning for a symbiotic UAV and UGV system for precision agriculture," *IEEE transactions on robotics*, vol. 32, no. 6, pp. 1498-1511, 2016, doi: 10.1109/iros.2013.6697126.

[3] R. Bonatti, Y. Zhang, S. Choudhury, W. Wang, and S. Scherer, "Autonomous Drone Cinematographer: Using Artistic Principles to Create Smooth, Safe, Occlusion-Free Trajectories for Aerial Filming," *Proceedings of the 2018 International Symposium on Experimental Robotics*, pp. 119–129, 2020, doi: 10.1007/978-3-030-33950-0_11.

[4] S. S. Mansouri, C. Kanellakis, D. Kominiak, and G. Nikolakopoulos, "Deploying MAVs for autonomous navigation in dark underground mine environments," *Robotics and Autonomous Systems*, vol. 126, p. 103472, Apr. 2020, doi: 10.1016/j.robot.2020.103472.

[5] A. Sahoo, S. K. Dwivedy, and P. S. Robi, "Advancements in the field of autonomous underwater vehicle," *Ocean Engineering*, vol. 181, pp. 145–160, Jun. 2019, doi: 10.1016/j.oceaneng.2019.04.011.

[6] R. C. Cardoso *et al*., "A Review of Verification and Validation for Space Autonomous Systems," *Current Robotics Reports*, vol. 2, no. 3, pp. 273–283, Jun. 2021, doi: 10.1007/s43154-021-00058-1.

[7] M. Z. A. Rashid, M. F. M. Yakub, S. A. Z. bin Shaikh Salim, N. Mamat, S. M. S. M. Putra, and S. A. Roslan, "Modeling of the in-pipe inspection robot: A comprehensive review," *Ocean Engineering*, vol. 203, p. 107206, May 2020, doi: 10.1016/j.oceaneng.2020.107206.

[8] W. E. Dixon, D. M. Dawson, E. Zergeroglu, and A. Beha. *Nonlinear Control of Wheeled Mobile Robots*. Lecture Notes in Control and Information Sciences, 2001, doi: 10.1007/bfb0113116.

[9] O. Y. Ismael, M. Qasim, M. N. Noaman, and A. Kurniawan, "Salp Swarm Algorithm-Based Nonlinear Robust Control of Magnetic Levitation System Using Feedback Linearization Approach," *Proceedings of the 3rd International Conference on Electronics, Communications and Control Engineering*, pp. 58-64, Apr. 2020, doi: 10.1145/3396730.3396734.

[10] M. N. Alghanim, M. Qasim, K. P. Valavanis, M. J. Rutherford, and M. Stefanovic, "Comparison of Controller Performance for UGV-Landing Platform Self-Leveling," *2020 28th Mediterranean Conference on Control and Automation (MED)*, pp. 471-478, Sep. 2020, doi: 10.1109/med48518.2020.9182837.

[11] O. Y. Ismael, M. Qasim, and M. N. Noaman, "Equilibrium Optimizer-Based Robust Sliding Mode Control of Magnetic Levitation System," *Journal Européen des Systèmes Automatisés*, vol. 54, no. 1, pp. 131–138, Feb. 2021, doi: 10.18280/jesa.540115.

[12] M. N. Alghanim, M. Qasim, K. P. Valavanis, M. J. Rutherford, and M. Stefanovic, "Passivity-Based Adaptive Controller for Dynamic Self-Leveling of a Custom-Built Landing Platform on Top of a UGV," *2020 28th Mediterranean Conference on Control and Automation (MED)*, pp. 458-464, Sep. 2020, doi: 10.1109/med48518.2020.9182807.

[13] Mohd. N. Zafar and J. C. Mohanta, "Methodology for Path Planning and Optimization of Mobile Robots: A Review," *Procedia Computer Science*, vol. 133, pp. 141–152, 2018, doi: 10.1016/j.procs.2018.07.018.

[14] S. A. Bonab and A. Emadi, "Optimization-based Path Planning for an Autonomous Vehicle in a Racing Track," *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, pp. 3823-3828, Oct. 2019, doi: 10.1109/iecon.2019.8926856.

[15] N. D. Potdar, G. C. H. E. de Croon, and J. Alonso-Mora, "Online trajectory planning and control of a MAV payload system in dynamic environments," *Autonomous Robots*, vol. 44, no. 6, pp. 1065–1089, Jun. 2020, doi: 10.1007/s10514-020-09919-8.

[16] M. Hoy, A. S. Matveev, and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 33, no. 3, pp. 463–497, Mar. 2014, doi: 10.1017/s0263574714000289.

[17] M. W. Mehrez, K. Worthmann, J. P. V. Cenerini, M. Osman, W. W. Melek, and S. Jeon, "Model Predictive Control without terminal constraints or costs for holonomic mobile robots," *Robotics and Autonomous Systems*, vol. 127, p. 103468, May 2020, doi: 10.1016/j.robot.2020.103468.

[18] F. Xie and R. Fierro, "First-state contractive model predictive control of nonholonomic mobile robots," *2008 American Control Conference*, pp. 3494-3499, Jun. 2008, doi: 10.1109/acc.2008.4587034.

[19] T. Ding, Y. Zhang, G. Ma, Z. Cao, X. Zhao, and B. Tao, "Trajectory tracking of redundantly actuated mobile robot by MPC velocity control under steering strategy constraint," *Mechatronics*, vol. 84, p. 102779, Jun. 2022, doi: 10.1016/j.mechatronics.2022.102779.

[20] D. Wang, W. Wei, Y. Yeboah, Y. Li, and Y. Gao, "A Robust Model Predictive Control Strategy for Trajectory Tracking of Omni-directional Mobile Robots," *Journal of Intelligent & Robotic Systems*, vol. 98, no. 2, pp. 439–453, Dec. 2019, doi: 10.1007/s10846-019-01083-1.

[21] G. C. Karras and G. K. Fourlas, "Model Predictive Fault Tolerant Control for Omni-directional Mobile Robots," *Journal of Intelligent & Robotic Systems*, vol. 97, no. 3–4, pp. 635–655, May 2019, doi: 10.1007/s10846-019-01029-7.

[22] T. P. Nascimento, C. E. T. Dórea, and L. M. G. Gonçalves, "Nonlinear model predictive control for trajectory tracking of nonholonomic mobile robots," *International Journal of Advanced Robotic Systems*, vol. 15, no. 1, Jan. 2018, doi: 10.1177/1729881418760461.

[23] X. Zhu, C. Ding, L. Jia, and Y. Feng, "Koopman operator based model predictive control for trajectory tracking of an omnidirectional mobile manipulator," *Measurement and Control*, vol. 55, no. 9–10, pp. 1067–1077, Aug. 2022, doi: 10.1177/00202940221095559.

[24] J. B. Rawlings and K. R. Muske, "The stability of constrained receding horizon control," *IEEE Transactions on Automatic Control*, vol. 38, no. 10, pp. 1512–1516, 1993, doi: 10.1109/9.241565.

[25] W. Esterhuizen, K. Worthmann, and S. Streif, "Recursive Feasibility of Continuous-Time Model Predictive Control Without Stabilising Constraints," *IEEE Control Systems Letters*, vol. 5, no. 1, pp. 265–270, Jan. 2021, doi: 10.1109/lcsys.2020.3001514.

[26] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control*. Communications and Control Engineering, 2017, doi: 10.1007/978-3-319-46024-6.

[27] S. A. Emami and A. Banazadeh, "Simultaneous trajectory tracking and aerial manipulation using a multi-stage model predictive control,"

*Aerospace Science and Technology*, vol. 112, p. 106573, May 2021, doi: 10.1016/j.ast.2021.106573.

[28] J. Köhler, M. A. Müller, and F. Allgöwer, "A nonlinear tracking model predictive control scheme for dynamic target signals," *Automatica*, vol. 118, p. 109030, Aug. 2020, doi: 10.1016/j.automatica.2020.109030.

[29] Z. Wang, J. Zhan, C. Duan, X. Guan, P. Lu and K. Yang, "A Review of Vehicle Detection Techniques for Intelligent Vehicles," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 8, pp. 3811-3831, Aug. 2023, doi: 10.1109/TNNLS.2021.3128968.

[30] O. H. Jafari, D. Mitzel, and B. Leibe, "Real-time RGB-D based people detection and tracking for mobile robots and head-worn cameras," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5636-5643, May 2014, doi: 10.1109/icra.2014.6907688.

[31] T. Eppenberger, G. Cesari, M. Dymczyk, R. Siegwart, and R. Dube, "Leveraging Stereo-Camera Data for Real-Time Dynamic Obstacle Detection and Tracking," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10528-10535, Oct. 2020, doi: 10.1109/iros45743.2020.9340699.

[32] M. Qasim and O. Y. Ismael, "Shared Control of a Robot Arm Using BCI and Computer Vision," *Journal Européen des Systèmes Automatisés*, vol. 55, no. 1, pp. 139–146, Feb. 2022, doi: 10.18280/jesa.550115.

[33] D. Kloeser, T. Schoels, T. Sartor, A. Zanelli, G. Prison, and M. Diehl, "NMPC for Racing Using a Singularity-Free Path-Parametric Model with Obstacle Avoidance," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 14324–14329, 2020, doi: 10.1016/j.ifacol.2020.12.1376.

[34] Y. Liu *et al*., "Robust nonlinear control approach to nontrivial maneuvers and obstacle avoidance for quadrotor UAV under disturbances," *Robotics and Autonomous Systems*, vol. 98, pp. 317–332, Dec. 2017, doi: 10.1016/j.robot.2017.08.011.

[35] B. Hermans, P. Patrinos, and G. Pipeleers, "A Penalty Method Based Approach for Autonomous Navigation using Nonlinear Model Predictive Control," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 234–240, 2018, doi: 10.1016/j.ifacol.2018.11.019.

[36] W. Li, C. Yang, Y. Jiang, X. Liu, and C.-Y. Su, "Motion Planning for Omnidirectional Wheeled Mobile Robot by Potential Field Method," *Journal of Advanced Transportation*, vol. 2017, pp. 1–11, 2017, doi: 10.1155/2017/4961383.

[37] G. Klančar and I. Škrjanc, "Tracking-error model-based predictive control for mobile robots in real time," *Robotics and Autonomous Systems*, vol. 55, no. 6, pp. 460–469, Jun. 2007, doi: 10.1016/j.robot.2007.01.002.

[38] G. V. Raffo, G. K. Gomes, J. E. Normey-Rico, C. R. Kelber, and L. B. Becker, "A Predictive Controller for Autonomous Vehicle Path Tracking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 1, pp. 92–102, Mar. 2009, doi: 10.1109/tits.2008.2011697.

[39] J. Backman, T. Oksanen, and A. Visala, "Navigation system for agricultural machines: Nonlinear Model Predictive path tracking," *Computers and Electronics in Agriculture*, vol. 82, pp. 32–43, Mar. 2012, doi: 10.1016/j.compag.2011.12.009.

[40] D. Gu and H. Hu, "A stabilizing receding horizon regulator for nonholonomic mobile robots," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 1022–1028, Oct. 2005, doi: 10.1109/tro.2005.851357.

[41] M. N. Noaman, M. Qasim, and O. Y. Ismael, "Landmarks exploration algorithm for mobile robot indoor localization using VISION sensor," *Journal of Engineering Science & Technology*, vol. 16, no. 4, pp. 3165-3184, 2021.

[42] J. Lin, H. Zhu, and J. Alonso-Mora, "Robust Vision-based Obstacle Avoidance for Micro Aerial Vehicles in Dynamic Environments," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2682-2688, May 2020, doi: 10.1109/icra40945.2020.9197481.

[43] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, "Model Predictive Contouring Control for Collision Avoidance in Unstructured Dynamic Environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4459–4466, Oct. 2019, doi: 10.1109/lra.2019.2929976.

[44] M. Parimala, D. Lopez, and N. Senthilkumar, "A survey on density based clustering algorithms for mining large spatial databases," *International Journal of Advanced Science and Technology*, vol. 31, no. 1, pp. 59-66, 2011.

[45] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *Autonomous Robot Vehicles*, pp. 396–404, 1986, doi: 10.1007/978-1-4613-8997-2_29.

[46] F. Gao, W. Wu, Y. Lin, and S. Shen, "Online Safe Trajectory Generation for Quadrotors Using Fast Marching Method and Bernstein Basis Polynomial," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 344-351, May 2018, doi: 10.1109/icra.2018.8462878.

[47] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, Mar. 1997, doi: 10.1109/100.580977.

[48] F. Gao, Y. Lin, and S. Shen, "Gradient-based online safe trajectory generation for quadrotor flight in complex environments," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3681-3688, Sep. 2017, doi: 10.1109/iros.2017.8206214.

[49] G. Ferrer, A. Garrell, and A. Sanfeliu, "Robot companion: A social-force based approach with human awareness-navigation in crowded environments," *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1688-1694, 2013, doi: 10.1109/IROS.2013.6696576.

[50] B. T. Lopez and J. P. How, "Aggressive 3-D collision avoidance for high-speed navigation," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5759-5765, May 2017, doi: 10.1109/icra.2017.7989677.

[51] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, Jun. 2017, doi: 10.1177/0278364917712421.

[52] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-Based Model Predictive Control: Toward Safe Learning in Control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 269–296, May 2020, doi: 10.1146/annurev-control-090419-075625.

[53] J. Berberich, J. Köhler, M. A. Müller and F. Allgöwer, "Data-Driven Model Predictive Control With Stability and Robustness Guarantees," in *IEEE Transactions on Automatic Control*, vol. 66, no. 4, pp. 1702-1717, April 2021, doi: 10.1109/TAC.2020.3000182.

[54] R. Carli, G. Cavone, N. Epicoco, P. Scarabaggio, and M. Dotoli, "Model predictive control to mitigate the COVID-19 outbreak in a multi-region scenario," *Annual Reviews in Control*, vol. 50, pp. 373-393, 2020, https://doi.org/10.1016/j.arcontrol.2020.09.005.

[55] J. Zeng, B. Zhang, and K. Sreenath, "Safety-Critical Model Predictive Control with Discrete-Time Control Barrier Function," *2021 American Control Conference (ACC)*, pp. 3882-3889, May 2021, doi: 10.23919/acc50511.2021.9483029.

[56] Z. Jian *et al*., "Dynamic Control Barrier Function-based Model Predictive Control to Safety-Critical Obstacle-Avoidance of Mobile Robot," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3679-3685, May 2023, doi: 10.1109/icra48891.2023.10160857.

[57] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, Jul. 2018, doi: 10.1007/s12532-018-0139-4.

[58] A. Wächter and L. T. Biegler, "Line Search Filter Methods for Nonlinear Programming: Motivation and Global Convergence," *SIAM Journal on Optimization*, vol. 16, no. 1, pp. 1–31, Jan. 2005, doi: 10.1137/s1052623403426556.

[59] J. C. L. Barreto S., A. G. S. Conceicao, C. E. T. Dorea, L. Martinez, and E. R. de Pieri, "Design and Implementation of Model-Predictive Control With Friction Compensation on an Omnidirectional Mobile Robot," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 2, pp. 467–476, Apr. 2014, doi: 10.1109/tmech.2013.2243161.

[60] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[61] W. B. Dunbar and R. M. Murray, "Model predictive control of coordinated multi-vehicle formations," *Proceedings of the 41st IEEE Conference on Decision and Control*, vol. 4, pp. 4631-4636, 2002, doi: 10.1109/cdc.2002.1185108.

[62] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, Jun. 2000, doi: 10.1016/s0005-1098(99)00214-9.

[63] J. Santos, A. G. S. Conceição, and T. L. M. Santos, "Trajectory tracking of Omni-directional Mobile Robots via Predictive Control plus a

Filtered Smith Predictor," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 10250–10255, Jul. 2017, doi: 10.1016/j.ifacol.2017.08.1286.

[64] J. Cenerini, M. W. Mehrez, J. Han, S. Jeon, and W. Melek, "Model Predictive Path Following Control without terminal constraints for holonomic mobile robots," *Control Engineering Practice*, vol. 132, p. 105406, Mar. 2023, doi: 10.1016/j.conengprac.2022.105406.

[65] H.-S. Kang, Y.-T. Kim, C.-H. Hyun, and M. Park, "Generalized Extended State Observer Approach to Robust Tracking Control for Wheeled Mobile Robot with Skidding and Slipping," *International Journal of Advanced Robotic Systems*, vol. 10, no. 3, p. 155, Jan. 2013, doi: 10.5772/55738.

[66] M. Fnadi, F. Plumet, and F. Benamar, "Nonlinear Tire Cornering Stiffness Observer for a Double Steering Off-Road Mobile Robot," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7529-7534, May 2019, doi: 10.1109/icra.2019.8794047.

[67] M. Cui, R. Huang, H. Liu, X. Liu, and D. Sun, "Adaptive tracking control of wheeled mobile robots with unknown longitudinal and lateral slipping parameters," *Nonlinear Dynamics*, vol. 78, no. 3, pp. 1811–1826, Jul. 2014, doi: 10.1007/s11071-014-1549-0.