



Real-time path planning for autonomous vehicle based on teaching–learning-based optimization

Ahmed D. Sabiha¹ · Mohamed A. Kamel² · Ehab Said¹ · Wessam M. Hussein³

Received: 28 October 2021 / Accepted: 11 May 2022 / Published online: 27 June 2022
© The Author(s) 2022

Abstract

This paper presents an online path planning approach for an autonomous tracked vehicle in a cluttered environment based on teaching–learning-based optimization (TLBO), considering the path smoothness, and the potential collision with the surrounding obstacles. In order to plan an efficient path that allows the vehicle to be autonomously navigated in cluttered environments, the path planning problem is solved as a multi-objective optimization problem. First, the vehicle perception is fully achieved by means of inertial measurement unit (IMU), wheels odometry, and light detection and ranging (LiDAR). In order to compensate the sensors drift to achieve more reliable data and improve the localization estimation and corrections, data fusion between the outputs of wheels odometry, LiDAR, and IMU is made through extended Kalman filter (EKF). Then, TLBO is proposed and applied to determine the optimum online path, where the objectives are to find the shortest path to reach the target destination, and to maximize the path smoothness, while avoiding the surrounding obstacles, and taking into account the vehicle dynamic and algebraic constraints. To check the performance of the proposed TLBO algorithm, it is compared in simulation to genetic algorithm (GA), particle swarm optimization (PSO), and a hybrid GA–PSO algorithm. Finally, real-time experiments based on robot operating system (ROS) implementation are conducted to validate the effectiveness of the proposed path planning algorithm.

Keywords Autonomous tracked vehicles · Path planning · Teaching–learning-based optimization (TLBO)

1 Introduction

The ability to reach a desired target and to avoid the surrounding obstacles are vital aspects of autonomous vehicles'. The main idea of path planning is to get a sequence of points and

segments that allow the vehicle to navigate safely through the working space towards the desired destination. Besides, the path planning algorithm must consider the collision avoidance with the surrounding obstacles, and achieve some other goals such as minimizing the traveled distance, the mission time, the consumed energy, and/or any other goals based on the mission type.

There are two major techniques for path planning: (i) global or off-line path planning approach, and (ii) local or online path planning approach [1]. Typically, a global path planning produces a high-level low-resolution path based on knowing its current and previous perceptive environmental information, or what is so called the environmental map. This technique is successful in developing an optimized path. Nevertheless, the reaction to dynamic or unexpected obstacles is inadequate. Local path planning provides a minimum-level high-resolution path based on the gathered on-board sensory information. However, when the environment is cluttered or the objective is a long distance far, this approach is inefficient. Therefore, it is better to combine both approaches to

✉ Mohamed A. Kamel
mohamed.atef.kamel@mtc.edu.eg

Ahmed D. Sabiha
ahmeddesokyeg@gmail.com

Ehab Said
ehab_said@mtc.edu.eg

Wessam M. Hussein
wessam@eaeat-academy.edu.eg

¹ Department of Mechatronic Engineering, Military Technical College, Cairo 11766, Egypt

² Department of Mechanical Engineering, Military Technical College, Cairo 11766, Egypt

³ Department of Mechatronic Engineering, Egyptian Academy for Engineering and advanced Technology (EAEAT), Cairo, Egypt

improve their benefits and reduce some of their weaknesses [2].

Path planning navigational approaches can be classified into classical and bio-inspired approaches [3]. Most of the classical approaches were used before the 2000s, while later bio-inspired methods became to be the most dominant [4]. The basic concept of classical approaches is either to discover a feasible solution or to confirm that there is no solution. The main classical approaches are: cell decomposition [5,6], roadmap [7], sampling-based algorithms [8], and artificial potential field (APF) [9,10]. These methods are not usually mutually incompatible, and mostly a hybrid algorithm to contain two classical techniques is applied to improve the path planning process [11]. Although the classical approaches are simple, they have some drawbacks such as: high computational time, not being suitable for real-time implementation, and regional minima trapping. Consequently, bio-inspired approaches are proposed to avoid these drawbacks [3,12].

In this work, a real-time online path planning algorithm for a tracked unmanned ground vehicle (UGV) in an unknown and cluttered environment is addressed in this paper. First, the vehicle perception is achieved using the data fusion between the wheels' odometry, light detection and ranging (LiDAR), and inertial measurement unit (IMU). Then, teaching–learning-based optimization (TLBO) is proposed to find the near-optimum online path, where the objectives are to get the shortest path to reach the target destination, and to maximize the path smoothness, while avoiding the surrounding obstacles, and taking into consideration the vehicle dynamic and algebraic constraints.

Compared with the related works in the literature, the major contributions can be summarized as follows:

1. Developing a near-optimal online path planning algorithm based on TLBO, that obtains the desired collision-free path that the vehicle must follow in order to reach its destination; and
2. Implementation of the proposed algorithm in real-time to validate its applicability. The real-time experiments are performed based on robot operating system (ROS).

The rest of this paper is organized as follows. The related work is presented in Sect. 2. The formulation of the path planning problem as a multi-objective optimization problem is presented in Sect. 3. Next, TLBO approach is presented to solve the optimization problem in Sect. 4. Section 5 presents the simulation results of the proposed TLBO algorithm compared to genetic algorithm (GA), particle swarm optimization (PSO), and a hybrid GA–PSO. Section 6 illustrates the experimental results to validate the effectiveness and applicability of the presented approach. Finally, conclusions and future work recommendations are conducted in Sect. 7.

2 Related work

Most recently, computational intelligent approaches have become the most dominant in the field of unmanned systems' navigation due to their ability to handle the uncertainties in the robot's surrounding environment, and to find the near-optimal path considering the vehicle's algebraic and dynamic constraints, as well as any other constraints corresponding to the vehicle or its assigned mission. The main bio-inspired approaches used for UGVs path planning are: GA, fuzzy logic (FL), neural network (NN), PSO, firefly algorithm (FFA), and most recently, TLBO.

In [13], a path planning algorithm for a mobile robot is proposed based on GA to find a collision-free path. The proposed approach incorporates GA with a modified mutation operation as a solution technique to ensure a selection of grids from the available zones without any repetition. The proposed model tries to shrink the search space, which in turn reduces the computational complexity. In [14], navigation from narrow passages is investigated. The authors proposed a dependent behavior modulation combining motor schema and FL. Nevertheless, the scalability of this algorithm is its major demerit and results in a huge number of fuzzy rules in the case of a complicated behavior-based systems. Changwon et al. [15] presented a fuzzy analytic hierarchy process (FAHP) for navigation of a mobile robot to be efficiently handled by means of a decision-making multi-objective problem. The presented FAHP differs from traditional AHP in using triangulation of fuzzy number-based extent analysis to produce the weight vectors. In [16], a fast simultaneous localization and mapping technique (SLAM) based on NN is exploited to mitigate the error accumulation caused by the imprecise linearization of the SLAM nonlinear function, and the improper odometry model. Results showed the capability of a wheeled mobile robot (WMR) to safely navigate in an unknown and cluttered environment. In [17], an online path planner is based on NN, where the performance of this algorithm is estimated based on two offline training path data sets. Nevertheless, experimental results are omitted, and simulation results are obtained. In [18], the localization and mapping of a mobile robot during the navigation in an unknown environment is addressed. A multi-agent particle filter is applied to handle the mapping and localization challenges, where PSO is exploited to improve the computational burden and the convergence characteristics. To avoid the problems of premature convergence and local-minima trapping associated with PSO, an improved PSO algorithm is proposed in [19] to obtain a near-optimal path for mobile robots. Besides, the path smoothness is achieved based on a continuous high-degree Bézier curve. Fengling et al. [20] exploited an optimal path planning algorithm based on FFA with self-adaptive population size. Two nonlinear functions are presented to get the population size based on the degree of

a potential collision. Results showed that the proposed algorithm is stable and its convergence speed and computational time are acceptable.

It can be noted that all the bio-inspired approaches have some merits and suffer from other demerits. Therefore, many hybrid algorithms consist of two approaches are proposed to enhance the overall efficiency and performance of the path planner [21–24]. Also, all swarm and evolutionary techniques are random and probabilistic. As a result, they must have some common controlling parameters, such as the population size. Furthermore, each algorithm has its own specific control parameters such as the crossover and mutation rates in GA, and the inertia weight and cognitive and social parameters in PSO. It is worth noted that the proper tuning of the algorithm-specific parameters is a critical factor affecting the computational time and obtaining the feasible solution. Considering this fact, TLBO is proposed as it does not need any specific parameters.

TLBO is a meta-heuristic approach proposed by Rao et al. in 2011 [25]. It is inspired by the learning and teaching process, via a simplified mathematical model of knowledge enhancements gained by learners in the classroom [26]. In TLBO, the population is defined as a group of learners, while the most learned one is assigned as a teacher, i.e. the learner with the best results. Based on that, there are two stages of TLBO: the teacher stage and the learner stage. In the first stage, all learners improve their knowledge from the teacher. Then, in the second stage, learners continue learning and improving through peer interaction between them.

TLBO is widely proposed for path planning of robotic manipulators [27,28]. However, very few works use TLBO for UGV path planning. In [29], path planning algorithm for a mobile robot navigation is presented based on nonlinear inertia weighted TLBO (NIWTLBO). Based on a coordinate system transformation, a new map model is built. Then, a global path is obtained by means of NIWTLBO. The proposed algorithm performance is tested through simulation. However, experimental results are omitted. In [30], a path planner based on adaptive neural fuzzy inference system (ANFIS) is applied, where TLBO is implemented to find the main parameters of ANFIS. Compared with PSO, invasive weed optimization (IWO), and biogeography-based optimization (BBO), simulation results showed a better performance for the proposed algorithm. In [31], a conformal geometric algebra and TLBO are proposed for a mobile robot's path planning. First, the robot and the obstacles are considered as conformal spheres. Then, in order to convert the robot sphere to a new sphere location, a conformal translator is provided. Finally, TLBO algorithm is applied to optimize this translator by minimizing the distance between the desired position and the new position, in addition to maximize the distance between all possible obstacles positions and the new position. In [32], TLBO technique is proposed for

path planning to navigate a three-wheeled robot. The considered objective function is to safely reach the desired target. In [33], a path planning algorithm based on shuffled frog leaping algorithm (SFLA) is exploited. To improve the exploitation and increase the convergence rate of SFLA, a hybrid SLFA-TLBO algorithm is presented. Simulation results emphasize the applicability of this hybrid algorithm compared with the traditional TLBO and PSO.

From the literature review, the following results are summarized for the problem of unmanned tracked vehicles' path planning:

1. Classical approaches are simple. However, they have some disadvantages like high computational time, the difficulty of online implementation, and local minima trapping;
2. Bio-inspired approaches are able to find the near-optimal path. However, their main demerit is the tuning of the algorithm's control parameters;
3. TLBO is widely applied in path planning of robotic manipulators [27,28]. However, very few works focused on applying TLBO for UGV path planning [31–33]; and
4. Most of the existing TLBO techniques have not yet been validated experimentally.

3 Path planning problem

Let $q \in \mathcal{C}$ be the vector of generalized coordinates for the vehicle, $q = [x, y]^T$, and \mathcal{C} is the vehicle's configuration space. The vehicle moves in an unknown cluttered environment starting from S_p with coordinates (x_s, y_s) , while the goal is to reach the target location T_p with coordinates (x_t, y_t) as shown in Fig. 1. During motion, let q be the current posture of the vehicle's center of gravity with coordinates (x, y) and orientation angle θ , then:

$$q(t) = [x(t), y(t)], \quad q \in \mathbb{R}^2$$

In the vehicle configuration space \mathcal{C} , the start and the target points are linked with a line segment regardless the obstacles considered as the shortest path. This line is divided into $(M - 1)$ segments, and M points starting from P_1 to P_M , where every two successive points define a segment which is part of the path linking S_p to T_p . Consequently, to avoid the potential collision with the surrounding obstacles, and considering all other constraints, a new path consists of $(M - 1)$ segments and M points (including source and target locations) is planned.

Based on the aforementioned explanation, the objective of the path planning algorithm is to determine the optimum coordinates of $(M - 2)$ points (as S_p and T_p are already known) that minimizes the path length, maximizes the path smoothness, and avoids the potential collision with the surrounding obstacles. For this purpose, TLBO is proposed.

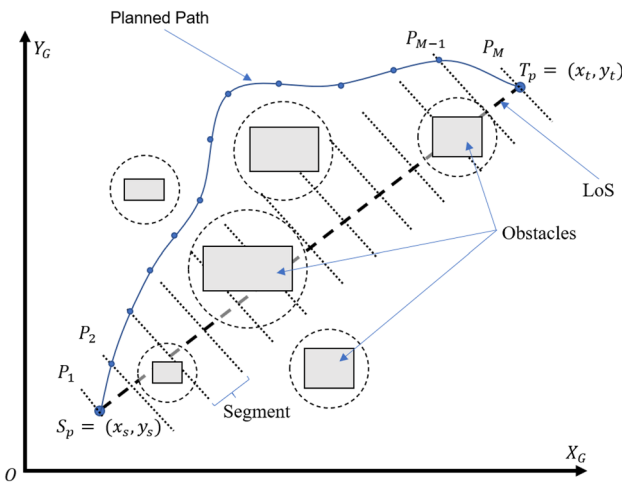


Fig. 1 Path planning modeling in 2D space

As illustrated before, the vector of vehicle position is defined as:

$$q = [x, y]^T, \quad q \in \mathbb{R}^2$$

As the path is divided into $(M - 1)$ segments, at the end of any segment, the vehicle position can be obtained as:

$$q_i = [x_i, y_i], \quad \forall i \in \{2, 3, \dots, (M - 1)\} \quad (1)$$

Remark 1 In this work, points P_1 and P_M are considered as the starting and target points S_p and T_p , respectively.

Consequently, the optimal path planning problem can be declared as: finding the optimum coordinates (x_i, y_i) and orientation angle $\theta_i, i \in \{2, 3, \dots, (M - 1)\}$ in order to minimize the path length and maximize the path smoothness, such that:

$$\begin{aligned} \min_u J(u), \\ J(u) = w_1 J_1(u) + w_2 J_2(u) \end{aligned} \quad (2)$$

where $J(u)$ is the total objective function, $J_1(u)$ is the objective function that minimizes the path length, $J_2(u)$ is the objective function that maximizes the path smoothness, w_1 and w_2 are weighting factors, and u is the vector of m design variables, such that:

$$u = [x_i, y_i], \quad \forall i \in \{2, 3, \dots, (M - 1)\}, \quad u \in \mathbb{R}^{2 \times (M-2)} \quad (3)$$

The values of J_1 and J_2 can be calculated as follows:

1. *Minimizing the path length:*

The total path length is determined by:

$$\begin{aligned} J_1(u) = S &= \sum_{i=1}^M S(P_i, P_{i+1}) \\ &= \sum_{i=1}^M \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \end{aligned} \quad (4)$$

where $S(P_i, P_{i+1})$ presents the distance between two successive points P_i and P_{i+1} .

2. *Maximizing the path smoothness:*

The smoothness of a path is a significant factor in vehicle path planning, as the vehicle should not abruptly alter its orientation. The path smoothness leads to lesser time consumption and energy. So, the smoothness is considered as a secondary concern. Consequently, maximizing path smoothness equals minimizing the total turning angle of the vehicle [34]. As a result, the following is the objective function of the vehicle’s turning angle:

$$J_2(u) = \Theta = \sum_{i=1}^{M-1} |\theta_i| \quad (5)$$

where θ_i is the angle between two vectors $\overrightarrow{P_{i-1}P_i}$ and $\overrightarrow{P_iP_{i+1}}$:

$$\begin{aligned} \theta_i &= \text{atan2}\left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i}\right) - \text{atan2}\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}}\right), \\ &\forall i \in \{2, 3, \dots, M\}. \end{aligned} \quad (6)$$

The optimization problem is subjected to the following constraints:

1. *The boundary constraints:*

In which the coordinates of P_i be always within the vehicle’s configuration space \mathcal{C} , i.e.

$$\{x_i, y_i\} \in \mathcal{C}, \quad \forall i \in \{2, 3, \dots, (M - 1)\} \quad (7)$$

2. *The collision avoidance constraint:*

During vehicle motion, the distance between the vehicle and the j^{th} obstacle, $j = \{1, 2, \dots, N_{obs}\}$, where N_{obs} is the number of detected obstacles, can be defined as:

$$\begin{aligned} d_j &= \sqrt{(x_i - x_{obs_j})^2 + (y_i - y_{obs_j})^2}, \\ &\forall i \in \{2, 3, \dots, (M - 1)\}, \forall j = \{1, 2, \dots, N_{obs}\} \end{aligned} \quad (8)$$

where obs_j denotes the center of gravity of the j th obstacle. As shown in Fig. 2, each obstacle j is surrounded by

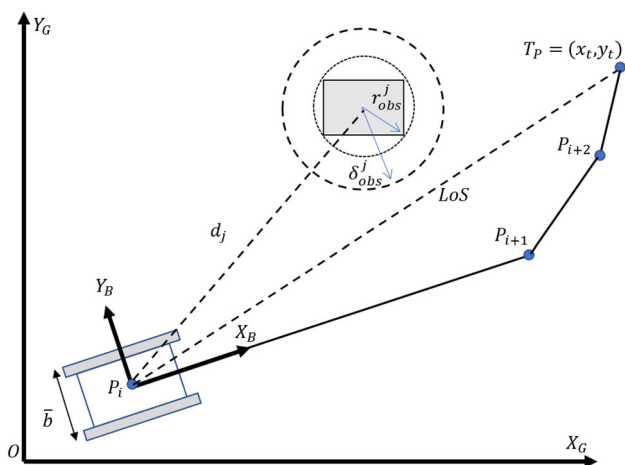


Fig. 2 The vehicle-obstacle interaction

a circle with radius r_{obs}^j . To prevent the collision between the vehicle and the j th obstacle, d_j must be greater than the minimum safety distance δ_{obs}^j presented as:

$$\delta_{obs}^j = r_{obs}^j + d_{safe} \tag{9}$$

where;

$$d_{safe} = \zeta \bar{b} \tag{10}$$

where ζ is a weighting factor, and \bar{b} is the tracked vehicle width.

Remark 2 The weighting factor ζ is a critical value to ensure the collision avoidance of the vehicle. Therefore, $0.5 < \zeta < 1$. The selection of ζ value is significantly critical. While increasing ζ will ensure collision avoidance, it will decrease the possibility of finding the feasible solution of the optimization problem, results in a high computational burden. In this work, ζ is selected to be $\frac{2}{3}$.

As a result, the collision avoidance constraint can be expressed as follows:

$$d_j \geq \delta_{obs}^j, \quad \forall j \in \{1, 2, \dots, N_{obs}\} \tag{11}$$

Finally, the path planning multi-objective optimization problem can be mathematically represented as follows:

$$\begin{aligned} & \min_u J(u), \\ & J(u) = w_1 S + w_2 \Theta \end{aligned} \tag{12}$$

subjected to:

$$\{x_i, y_i\} \in \mathcal{C}, \quad \forall i \in \{2, 3, \dots, (M - 1)\} \tag{13}$$

$$d_j \geq \delta_{obs}^j, \quad \forall j \in \{1, 2, \dots, N_{obs}\} \tag{14}$$

As TLBO is a stochastic technique, the above-mentioned constrained optimization problem presented in Eqs. (12) to (14) should be handled as an unconstrained problem, where all the constraints must be explicitly considered in the overall objective function. This can be achieved by adding penalties to the objective function for violating the constraints [35]. Generally, the objective function of a constrained optimization problem with c constraints can be expressed as an unconstrained problem based on the following general form:

$$J' = J + a_k \sum_{j=1}^c G_j[g_j(X)], \quad \forall j = \{1, 2, \dots, c\} \tag{15}$$

where G_j is the penalty function for the constraint g_j , and a_k is the penalty parameter, $a_k > 0$. According to Eq. (15), while minimizing the objective function, a positive penalty parameter is added whenever the constraint is violated [35]. There are various penalty functions. In this paper, the proposed penalty is proportional to the degree of the violation, where the constants a_k are adapted to adjust the contribution of the penalty terms to the objective function’s magnitude. As a result, Eq. (12) can be re-formulated as:

$$J(u) = w_1 S + w_2 \Theta + \sigma \sum_{j=1}^{N_{obs}} R_{obs}^j, \quad \forall j = \{1, 2, \dots, N_{obs}\} \tag{16}$$

where σ is the penalty coefficient, and R_{obs}^j can be calculated as follows:

$$R_{obs}^j = \begin{cases} 0 & \text{if } d_j \geq \delta_{obs}^j \\ \frac{1}{d_j} - \frac{1}{\delta_{obs}^j} & \text{if } d_j < \delta_{obs}^j \end{cases} \quad \forall j = \{1, 2, \dots, N_{obs}\} \tag{17}$$

meaning that if d_j is greater than δ_{obs}^j , i.e., the distance between the vehicle and the j th obstacle is safe. Therefore, the collision avoidance constraint is not violated, and no penalty is applied to the total objective function. However, if d_j is less than δ_{obs}^j , the vehicle will collide with the j th obstacle. As a result, a penalty is added to the objective function in order to avoid the potential collision.

Remark 3 Since the path is composed of line segments, it is differentiable everywhere, except at the unions between the line segments. Therefore, a high-degree Bézier curve is applied [19]. This will get the path differentiable everywhere, results in improving the path smoothness, and obtaining a suitable path for vehicle motion control.

4 TLBO Algorithm

After defining the optimization problem in Sect. 3, The TLBO algorithm is provided to obtain the near-optimum path that minimizes the total objective function presented in Eq. (15).

Teaching–learning is an important process in which each learner seeks to learn something from the other learner in order to improve themselves. TLBO was proposed by Rao et al. [25,36] and Rao and Patel [37], which simulates the phenomenon of teaching–learning in a classroom. In TLBO, two basic types of learning are employed: (i) from the teacher to learners (known as the teacher phase) and (ii) from the interaction between learners (known as the learner phase). In TLBO, a *population* consists of a group of students, where various subjects offered to the students are considered as the *design variables*. The *objective function* of the optimization problem is equivalent to the mean result of each student in all subjects. The best student with the best objective function's value is assigned as a *teacher*, while TLBO tries to improve the solution with each iteration.

Remark 4 During the following explanation of the TLBO algorithm, the subscripts i , j , and k refer to the iteration number, the subject, and the learner, respectively.

The following steps show the execution of the TLBO algorithm:

Step 1. *Construction of the design variables vector X (the vector of learners' subjects):* For each student k , the vector of D_v subjects (the number of design variables) is:

$$\begin{aligned} X &= [x_c, y_c], \quad \forall c \in \{2, 3, \dots, (M-1)\}, \\ X &\in \mathbb{R}^{2 \times (M-2)} \end{aligned} \quad (18)$$

As a result, the length of the vector of design variables is: $D_v = 2 \times (M-2)$.

Step 2. *Initialization:*

- Choose the number of learners N_p .
- Choose the maximum number of iterations i_{\max} .
- Set the iteration counter i to 1

Step 3. *Initial generation of random population:* Generate a random population of learners offered D_v subjects as follows:

$$\begin{aligned} X_{(k,j)}^0 &= X_j^{\min} + r_j \times (X_j^{\max} - X_j^{\min}), \\ k &= [1, 2, \dots, N_p], \quad j = [1, 2, \dots, D_v] \end{aligned} \quad (19)$$

where, r_j is a random number with a uniform distribution within the range $\in [0, 1]$, N_p denotes the number of learners in a class, and X_j^{\max} and X_j^{\min} represent the upper and lower boundaries of each design variable j , respectively.

Step 4. *Teacher Phase:* In this phase, the objective is to find the best student in the class to be the teacher. Then, all other students are considered as learners. The following are the steps of the teacher phase:

Step 4.1 *Evaluate the mean result ($M_{i,j}$):* The mean result of all learners on a particular subject j , is given as follows:

$$M_{i,j} = \frac{1}{N_p} \sum_{k=1}^{N_p} X_{(i,k,j)} \quad (20)$$

Step 4.2 *Evaluate the objective function:* For each learner k , evaluate the objective function J presented in Eq. (15).

Step 4.3 *Find the teacher ($X_i^{teacher}$):* Determine the best learner of the generation with its associated values of design variables to be the teacher $X_i^{teacher}$ in the present iteration i , which is the best solution (the lowest value of J) among the other learners. Under normal circumstances, the teacher is usually considered to be a learner with a high degree of learning ability to educate the other learners, so they can have better learning achievement. In accordance with the teacher's abilities, the teacher tried to increase the average learning achievements of learners in the subjects they taught.

Step 4.4 *Evaluate the difference mean:* Evaluate the difference between the current mean result $M_{i,j}$ and the teacher results in each subject j as follows:

$$\text{Dif_Mean}_{i,j} = r_j \times (X_i^{teacher} - T_f M_{i,j}) \quad (21)$$

where r_j is a random number with a uniform distribution within the range $\in [0, 1]$, and T_f is the teaching factor, $T_f \in [1, 2]$ [26], which is determined randomly as follows:

$$T_f = \text{round}[1 + \text{rand}(0, 1)\{2 - 1\}] \quad (22)$$

Step 4.5 *Update the design variables' values:* For each learner, and based on $\text{Dif_Mean}_{i,j}$, the vector of design variables X is updated using the following formula:

$$X_{(i,k,j)}^{new} = X_{(i,k,j)} + \text{Dif_Mean}_{i,j} \quad (23)$$

Step 4.6 *Update the solution:* Recalculate the objective function according to the updated values $X_{(i,k,j)}^{new}$.

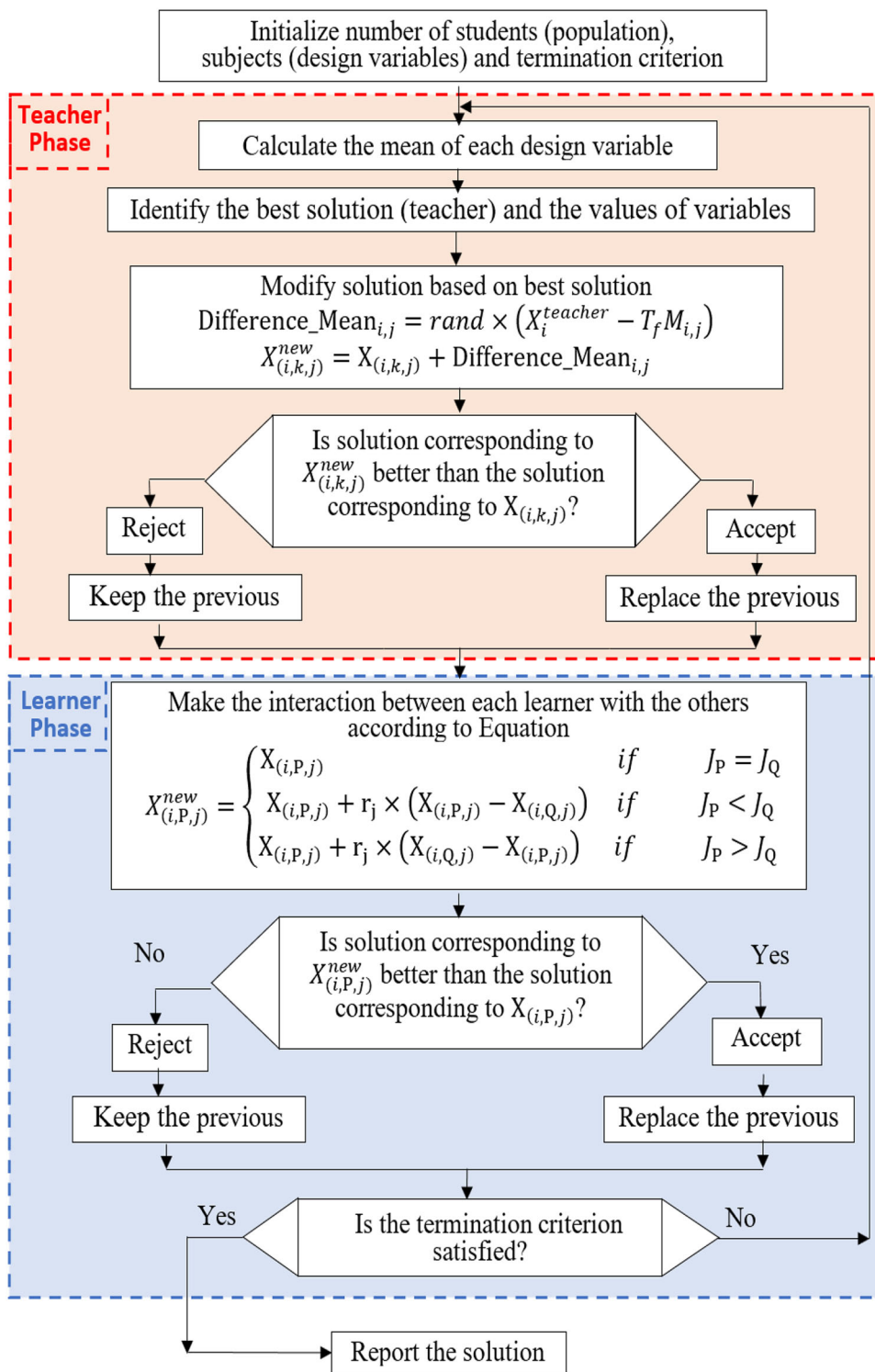
If $X_{(i,k,j)}^{new}$ yields a better solution than $X_{(i,k,j)}$, then $X_{(i,k,j)}^{new}$ replaces $X_{(i,k,j)}$. Otherwise, $X_{(i,k,j)}$ is maintained, i.e.,:

$$X_{i,k,j} = \begin{cases} X_{(i,k,j)} & \text{if } J(X_{(i,k,j)}^{new}) > J(X_{i,k,j}) \\ X_{(i,k,j)}^{new} & \text{if } J(X_{(i,k,j)}^{new}) < J(X_{i,k,j}) \end{cases} \quad (24)$$

This means that the knowledge of each learner is improved from the interaction with the teacher. By the end of this step, the teacher phase is ended, and the learning phase will start.

Step 5. Learner phase: During this phase, each learner interacts with other learners randomly to improve

Fig. 3 TLBO flow chart



their knowledge. The followings are the steps of the learner phase:

Step 5.1 Interaction: Consider two learners P and Q , the interaction between them to improve their knowledge can be obtained as follows:

$$X_{(i,P,j)}^{new} = \begin{cases} X_{(i,P,j)} & \text{if } J_P = J_Q \\ X_{(i,P,j)} + r_j \times (X_{(i,P,j)} - X_{(i,Q,j)}) & \text{if } J_P < J_Q \\ X_{(i,P,j)} + r_j \times (X_{(i,Q,j)} - X_{(i,P,j)}) & \text{if } J_P > J_Q \end{cases} \quad (25)$$

Step 5.2 Select the best values for each learner after interaction: Re-evaluate the objective function according to the updated values provided from Eq. (25). Similar to Eq. (24), if the new solution associated to $X_{(i,k,j)}^{new}$ is better than the current one $X_{(i,k,j)}$, then $X_{(i,k,j)}$ is replaced with $X_{(i,k,j)}^{new}$. Otherwise, $X_{(i,k,j)}$ is preserved. Learners who have been accepted are advanced to the next iteration.

Step 5.3 Check that the new solutions are within the boundary constraints: Check that $X_j^{min} \leq X_{(i,k,j)} \leq X_j^{max}$ as follows:

$$\begin{aligned} X_{(i,k,j)} &= X_j^{max} && \text{if } X_{(i,k,j)} > X_j^{max} \\ X_{(i,k,j)} &= X_j^{min} && \text{if } X_{(i,k,j)} < X_j^{min} \end{aligned} \quad (26)$$

Step 6. Termination criterion: Check for convergence of the iterative process after each iteration. If the termination criterion is met, terminate the algorithm. Otherwise, set $i = i + 1$ and, go to Step 4.

Remark 5 The termination criterion used in this study is that the optimization algorithm is ended if the objective function does not change after 30 iterations.

The flowchart of the TLBO procedure is shown in Fig. 3.

5 Simulation results analysis

In simulation, the proposed path planning algorithm is validated. The goal of the simulation is to prove that the presented path planning algorithm can find the near-optimal global offline obstacle-free path, and to check the stability and performance of the presented TLBO approach compared to GA, PSO and a hybrid GA-PSO algorithm [38]. All the parameters of GA and PSO are tuned based on [39].

Assumption 1 In simulation, it is assumed that all obstacles are already detected, and the vehicle’s workspace is completely known.

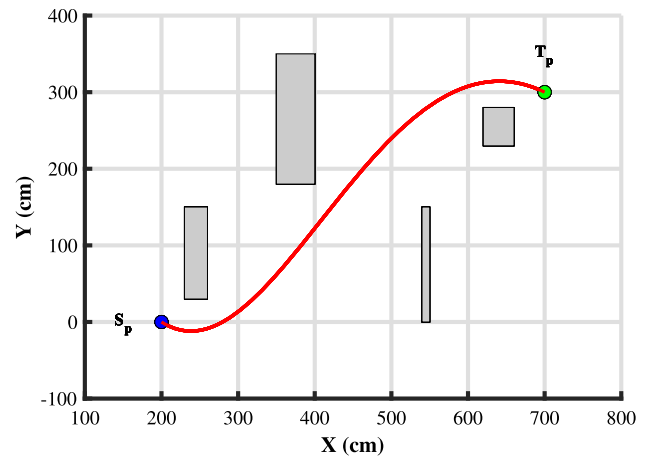


Fig. 4 Path planning trajectory in XY plane during simulation

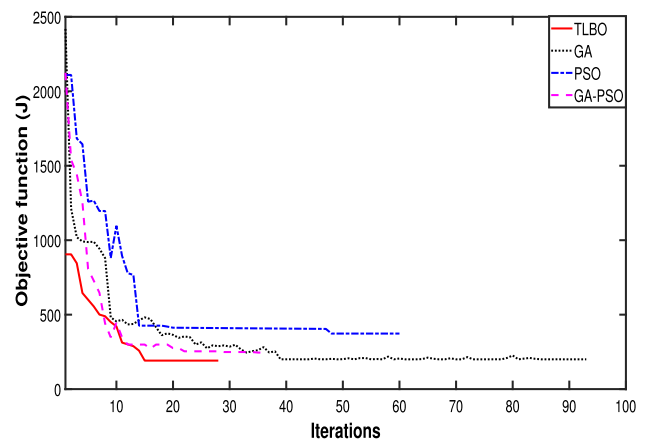


Fig. 5 The values of the objective function along the iterations

5.1 Path planning results

In simulation, the vehicle’s start point $S_p = (200, 0)$, and the target location is $T_p = (700, 300)$. Also, there are four detected obstacles in the configuration space with known locations, while the safety distance for each obstacle is $d_{safe} = 30$ cm.

As can be seen from Fig. 4, the presented algorithm succeeded to generate a planned path with avoiding the collision with surrounding obstacles. Also, Fig. 5 shows the convergence rate of the objective function in cases of TLBO, GA, PSO, and GA-PSO. It can be seen that TLBO is better than the other techniques as it converges in 28 iterations to an objective function value below the values obtained by GA-PSO, PSO, and GA, that converge in 37, 60, and 92 iterations, respectively.

5.2 TLBO performance and stability analysis

Since TLBO is a meta-heuristic optimization technique, the first iteration is based on a random population, it is not assured that the global optimum solution will be obtained. As a result, it is of great importance to test its performance and stability. This can be obtained by executing the proposed algorithm many times to ensure that the result is close to the optimal result. The proposed algorithm is executed 50 times, and results are shown in Table 1. As can be seen, the proposed TLBO algorithm is better than GA, PSO, and GA-PSO as it converges to the minimum value of the objective function in a smaller number of iterations, results in less computational time. Also, from the point-of-view of stability, TLBO is more stable as the standard deviation value in the case of TLBO is less than that in case of GA, PSO, and GA-PSO.

5.3 Evaluation of trajectory quality

To evaluate the quality of the planned trajectory, the proposed algorithm is executed 25 times, and the values of J_1 and J_2 are evaluated in case of TLBO, GA, PSO, and GA-PSO, as J_1 and J_2 can be considered as the indicators for the quality of the trajectory. As can be seen from Fig. 6, TLBO always finds the minimum values of J_1 and J_2 compared to the other optimization techniques. Also, it can be observed from Table 2 that TLBO is more stable as the standard deviation (STD) for the values of J_1 and J_2 is less than those obtained by GA, PSO, and GA-PSO.

5.4 Effect of weights w_1 and w_2

As mentioned in Sect. 3, the multi-objective optimization problem is converted to a single objective one by means of weight aggregation. So, it is important to study the effect of weights on the objective function. This is achieved by means of the multivariate partial least square (PLS) method [40]. The proposed algorithm is executed 38 times, where w_1 and w_2 are changing in each execution, then J_1 , J_2 , and J are evaluated. The PLS model is fitted by cross-validation to get a two-component model with $R_X = 0.97$ and $Q_X = 0.93$. R_X is known as the goodness of fit which is an estimate of the explanation ability of the model, where Q_X is known as goodness of prediction which is an estimate of the predictive ability of the model. Figure 7 demonstrates the scatter plot of the two score vectors, which indicates the distribution of the data results during the 38 executions. More analysis is conducted by examining the loading plots to investigate the relationships between different variables. Figure 8 shows the loading plot, clarifying the effect of the weights on the total objective function J . It can be noticed that w_1 has a remarkable effect on the total objective function J . A sensitivity analysis was conducted on w_1 , w_2 , J_1 , and J_2 in an attempt

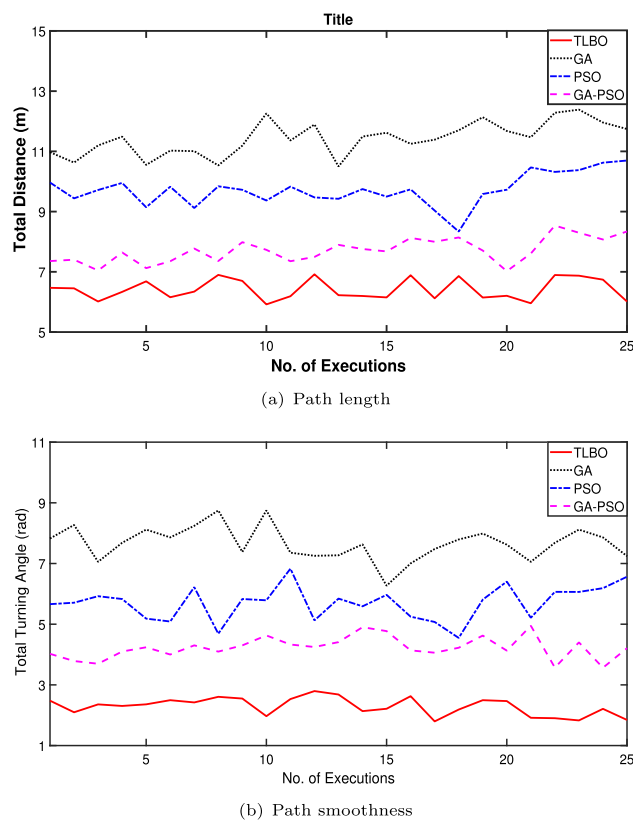


Fig. 6 Evaluation of the planned trajectory quality

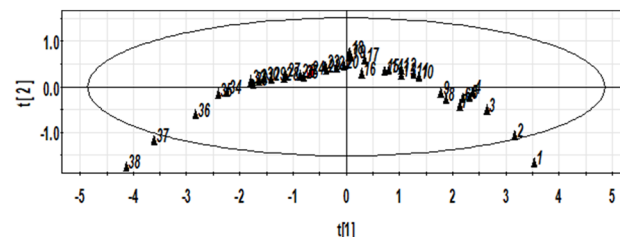


Fig. 7 Scatter plot of the PLS model

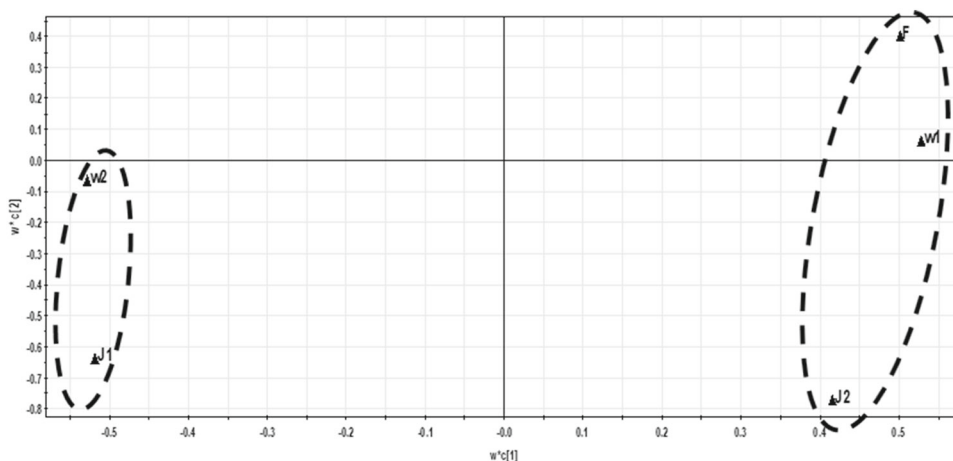
to extract the most significant features along the model on the total objective function J . Figure 9 shows the variable importance plot (VIP) of all available parameters. The influence of J_1 and w_1 along the data set is clearly noticed. Finally, in Fig. 10, a Pareto front is obtained. Based on this, the weights w_1 and w_2 are selected to be 0.65, 0.35, respectively.

6 Experimental results analysis

6.1 Platform description

To validate the proposed path planning algorithm presented in this study, the real-time analysis of the path planning is implemented in a cluttered environment, where the vehicle

Fig. 8 PLS loading plot



is surrounded by different obstacles. The experiment is conducted on a plane surface and is implemented on a small tracked vehicle shown in Fig. 11. All the vehicle parameters are obtained in Table 3.

The vehicle is powered by ROS. As shown in Fig. 11, it is driven by two EMG49 DC motors with wheel encoders. A power amplifier MD49 motor drive board is used as the motors controller and driver. A sensor fusion is used to improve the accuracy of the odometry by means of an IMU

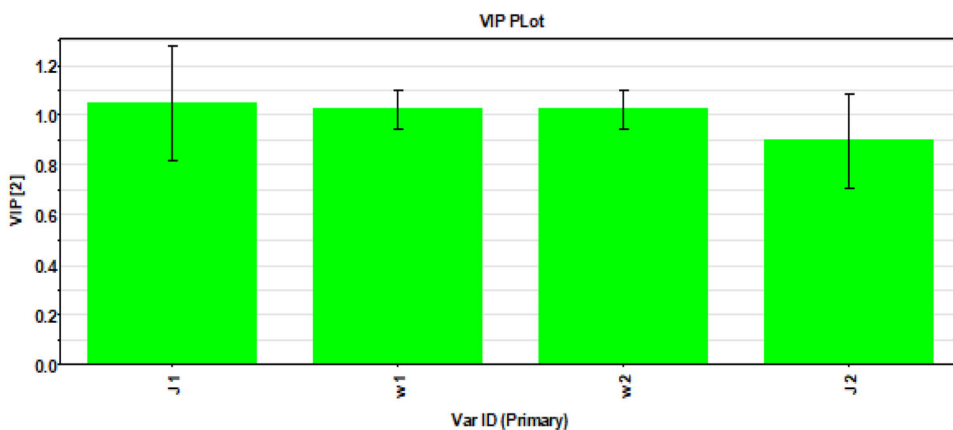
Table 1 Performance and stability analysis for TLBO, GA, PSO, and GA-PSO

Method	Average number of iterations	Average execution time (sec.)	Objective function		
			Best	Mean	STD
TLBO	32	22	911.565	1200.532	0.3462
GA	98	75	2425.588	2532.741	1.4232
PSO	66	60	2227.856	2328.348	0.935
GA-PSO	40	43	2119.588	2216.975	0.5186

Table 2 Results analysis for the trajectory quality indicators for TLBO, GA, PSO, and GA-PSO

	Path length				Path smoothness			
	Minimum	Maximum	Average	STD	Minimum	Maximum	Average	STD
TLBO	6.1184	6.9169	6.4134	0.3458	1.7966	2.7955	2.2907	0.2931
GA	10.4947	12.3855	11.4281	0.8645	4.5475	6.8299	5.6979	0.8203
PSO	8.3372	10.6960	9.7188	0.6239	6.2679	8.7503	7.6623	0.6902
GA-PSO	7.0216	8.5360	7.7117	0.4103	3.5645	4.9489	4.2280	0.3652

Fig. 9 VIP plot



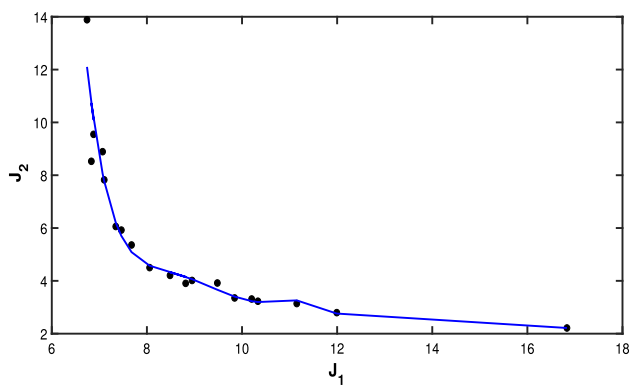


Fig. 10 Pareto front of the multi-objective optimization problem

Table 3 Vehicle physical parameters

Vehicle parameters	Value
Vehicle mass (m)	10kg
Track gauge (b)	39cm
Ground contact length of the track (L)	41.5cm
Track width (t_w)	7.5cm
Sprocket radius (r)	3.7cm

9-axes MPU9250. As the vehicle control is based on ROS, it consists of an off-board master computer and an on-board client computer. The off-board master PC has a processor Intel Core i5 CPU 3230M, 4GB RAM, and Linux Ubuntu 18.04 operating system (OS). While the on-board client computer is Raspberry-Pi 4 mounted on the vehicle, and has Linux Raspbian OS. It communicated with the MD49 motor driver by a TTL level serial bus, and with the IMU via I2C protocol. In addition, a LiDAR is mounted on the top of the vehicle to build a map of the surrounding environment. The LiDAR is connected to the on-board Raspberry Pi4 via USB connection.

The proposed ROS framework consists of packages and the nodes to interface the proposed controllers with the ROS environment, while the controllers, localization nodes and other components are implemented as C++ and Python libraries. The ROS framework allows the tracked vehicle to contribute and synchronize messages between nodes in the off-board master PC and with the client PC by using a network as shown in Fig. 12.

Before path planning, The vehicle navigation system has to solve two different problems: mapping and localization. After that, a path planning algorithm can be applied. To achieve the navigation task, the navigation nodes are designed to integrate the mapping and localization together. In order to build the map of the workspace and, get the instantaneous location of the vehicle, the navigation nodes received the information from wheels odometry, IMU, and LiDAR. Also, it received the location of the target position. Then, the

path planning algorithm can be implemented to generate the near-optimal path as required velocity and pose, Finally, the controller nodes produce the safe velocity commands and send it to the vehicle. The data fusion between the outputs of wheels odometry, LiDAR, and IMU is made through EKF in order to compensate the sensors drift to achieve more reliable data to improve the localization estimation and corrections.

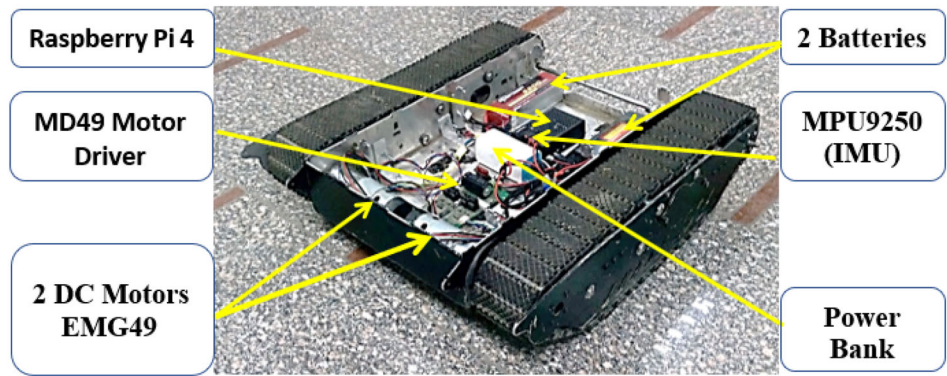
It’s worth noting that navigation nodes don’t need a prior static map. In fact, it can be started with or without a map. When started without a prior map, and based on the vehicle’s on-board sensors, the vehicle will have online information about the surrounded obstacles. Consequently, the vehicle will avoid the detected obstacles. At this stage, the vehicle will generate a global path for unknown places, which may collide with undiscovered obstacles. When the vehicle receives more information from its sensors about these unknown places, it will be able to re-plan its path.

Fig. 13 demonstrates the ROS transformation tree frames diagram `rqt_tf_tree` for path planning, where the transformation tree system builds the dynamic transformation that relates all reference frames in the system (drivers, IMU, and LiDAR) related to the frame body named base link.

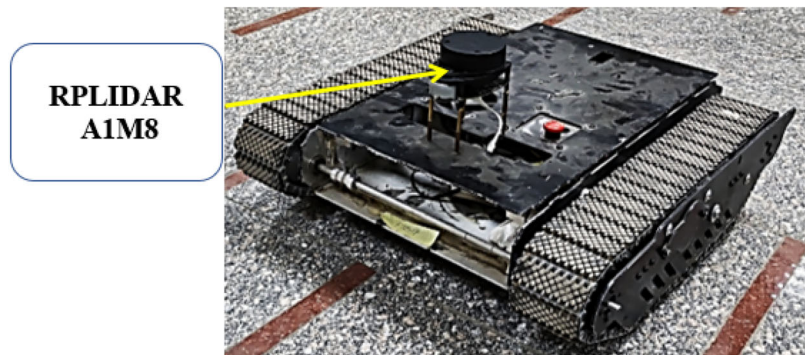
According to the ROS computation framework shown in Fig. 14 that represents the ROS computation graph diagram `rqt_graph` for the proposed path planning TLBO algorithm. The odometry is provided by the `nav_msgs/Odometry` message through ROS, which stores an estimation of the vehicle’s position and velocity in free space in order to identify its location. To avoid any obstacles, the sensor data is sent through `sensor_msgs/LaserScan` or `sensor_msgs/PointCloud` messages over ROS. The destination point is delivered to the navigation nodes via `geometry_msgs/PoseStamped` message. The velocity commands are sent by the controller nodes using the `/cmd_vel` topic in `geometry_msgs/Twist` messages.

To create a map using ROS, the `/mapping` node is proposed to construct an occupancy grid map. The `/mapping` node is considered as the implementation of simultaneous localization and mapping (SLAM), a technique that constructs a map of an environment simultaneously with the vehicle motion. Obviously, during the vehicle navigation through the environment, it gets information from the environment via its sensors and develops a map for the vehicle’s configuration space as a result. The node `/rplidarNode` generates the laser range data `sensor_msgs/LaserScan`, the `/mapping` node is used to build 2D occupancy grid maps. So, the `/mapping` node is used to create a map of an unknown environment as well as update an existing map. This can be seen in Fig. 15 where the vehicle motion in the generated map is visualized via the visualization environment “Rviz”.

Fig. 11 The vehicle used in the experiments with the main control system components

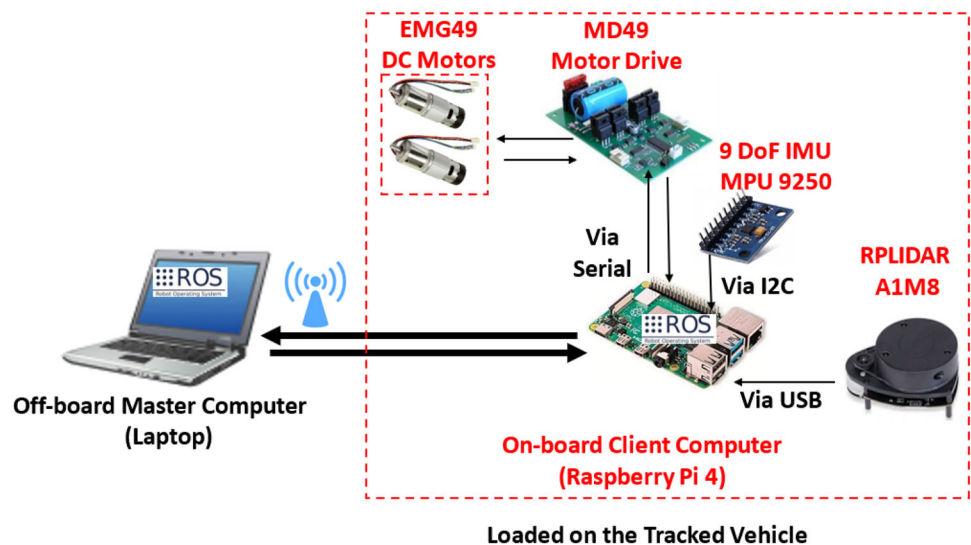


(a) The vehicle with internal components



(b) The vehicle with external components

Fig. 12 System architecture block diagram of the autonomous vehicle system



6.2 Results and analysis

In the experimental scenario, the start and target points are known where the vehicle moves in a cluttered environment, in which twelve random obstacles with different shapes and dimensions are existing as shown in Fig. 16. The configuration space dimensions are $(5m \times 5m)$, while the

coordinates of the start and target points are $S_p = (0, 0)$ and $T_p = (3.6, 3.9)$, respectively. All the values of the TLBO parameters are given in Table 4.

Figure 17 shows the experimental generated path. It can be seen that the vehicle reached the destination point, and evade the collision with the surrounding obstacles. As illustrated in Fig. 2, each obstacle is surrounded by two circles with radii

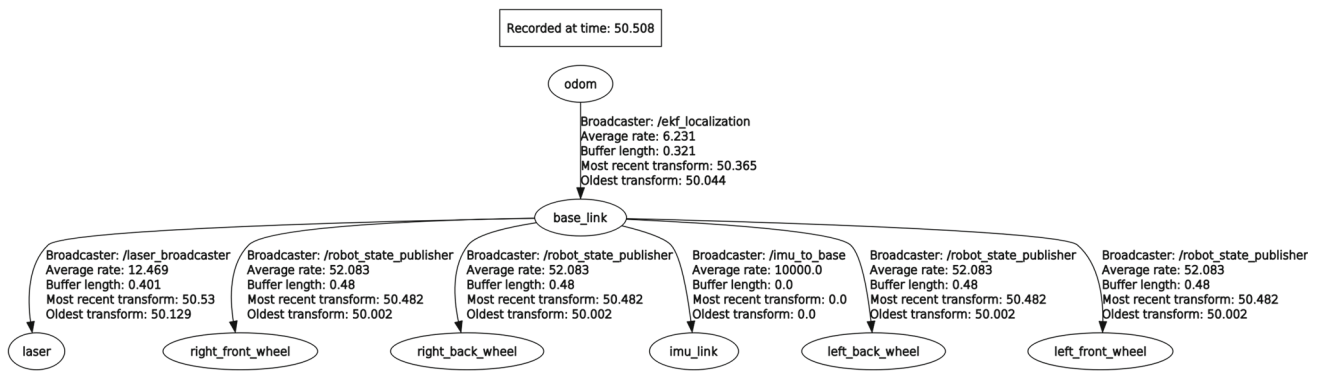


Fig. 13 ROS transformation tree frames diagram *rqt_tf_tree* for path planning

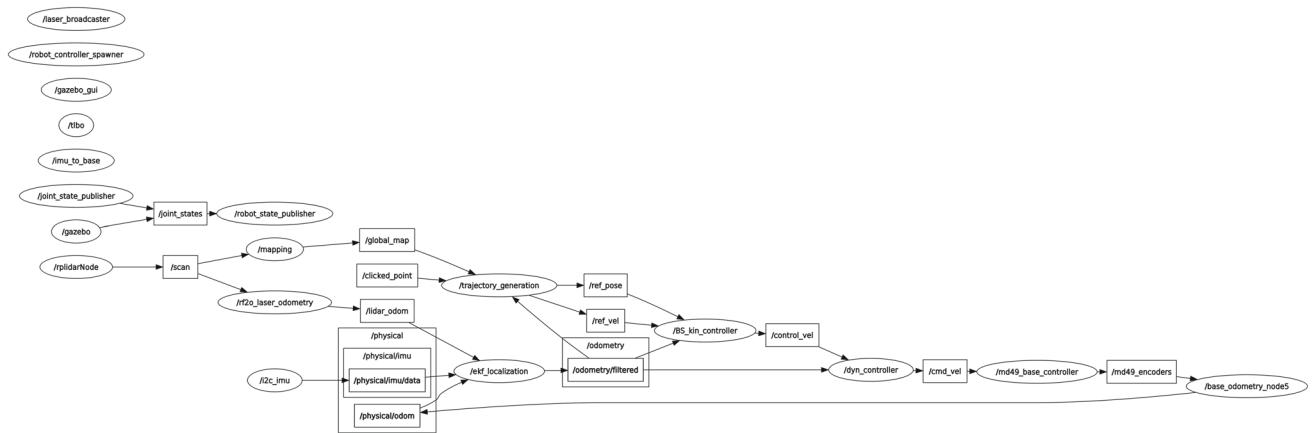


Fig. 14 ROS computation graph diagram *rqt_graph* for path planning

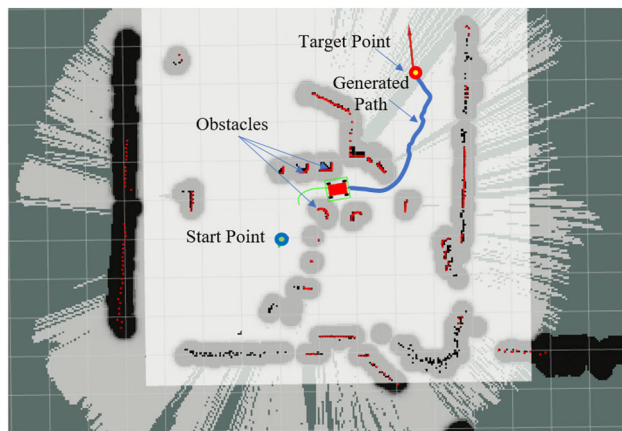


Fig. 15 The vehicle motion in the ROS-based visualization environment “Rviz”

r_{obs}^j and δ_{obs}^j . It can be noted from Fig. 17 that the generated path safely avoids the collision, as the path does not touch any of the circles with radius δ_{obs}^j .

As can be illustrated from Fig. 18, the presented TLBO algorithm can satisfy the collision avoidance constraint presented in Eq. (14), as it succeeded to ensure that the planned

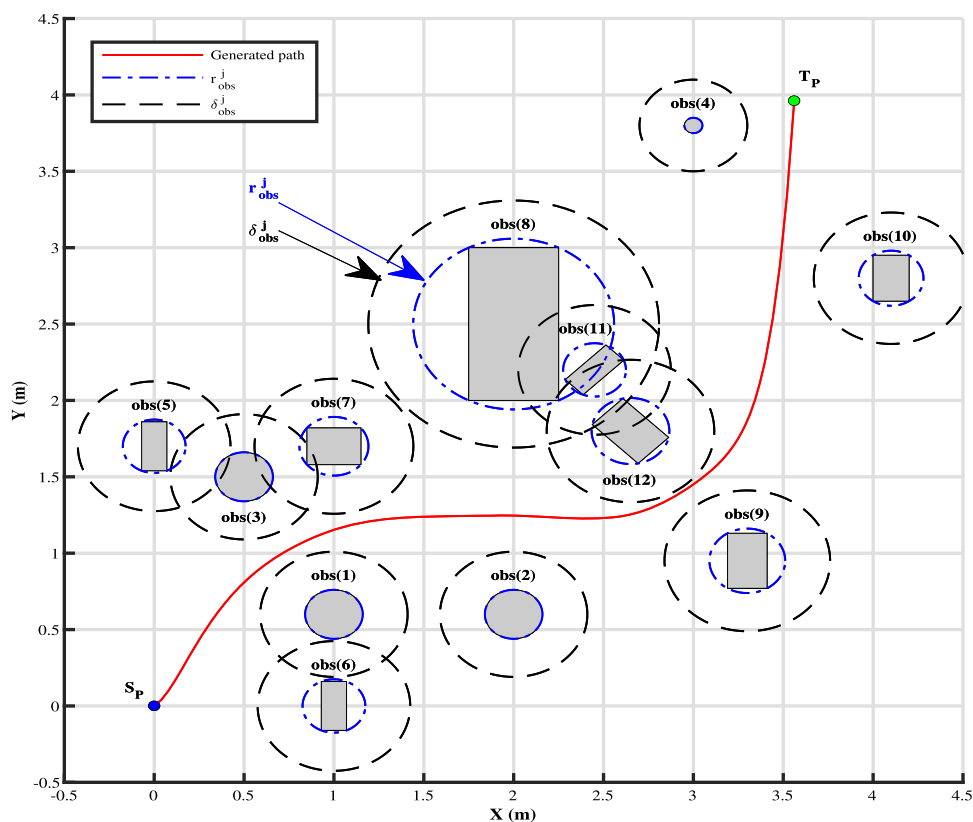
Table 4 TLBO parameters in experiment

Parameter	Value
Weighting factor (ζ)	2/3
Number of design variables (D_v)	2
Number of population (N_p)	100
Weighting value (w_1)	0.65
Weighting value (w_2)	0.35



Fig. 16 Configuration space of the real-time experiment

Fig. 17 Path planning vehicle trajectory with obstacles



path guarantees the safety of the vehicle by keeping a safe distance greater than δ_{obs}^j from the surrounding obstacles.

Figure 19 shows eight snapshots of the real-time experiment, where the tracked vehicle moves in the configuration space between the obstacles through a collision-free path during the experiment. At the top-right screen of each snapshot, the vehicle pose between the detected obstacles and the generated path is visualized in ROS/Rviz environment. The tracked vehicle analyzes the environment information from LiDAR, IMU, and driving motors' encoders through the sensor fusion by using EKF. The TLBO algorithm is implemented as a node to the ROS ecosystem by using C++ language to obtain the path with collision-free. The presented path planning TLBO algorithm is very effective since it recognizes and avoids obstacles in the surrounding environment, and it accomplishes the navigational task in a short time. Also, the computational time is measured to be 1.34 s for each frame that updates every 1.83 s.

7 Conclusions

This paper addresses the path planning algorithm for an autonomous tracked UGV in a cluttered environment based on TLBO algorithm. The proposed algorithm succeeded in obtaining the near-optimal paths with avoiding the sur-

rounding obstacles, and it can handle the problem of system kinematic and algebraic constraints. TLBO algorithm is applied to get the near-optimal path between the start and goal locations, considering the minimization of the path length, the maximization of the path smoothness, and avoiding the potential collision with the nearby obstacles. Many remarkable features are achieved by the proposed algorithm: (1) it is simple approach and can be implemented in real-time; and (2) using TLBO does not need the adjustment of specific parameters such as GA, FFA, PSO, and other meta-heuristic optimization techniques. Finally, the proposed path planning algorithm is experimentally evaluated in real-time in an indoor cluttered environment, and is implemented on an autonomous tracked vehicle based on ROS to emphasize the effectiveness of the presented path planning algorithm and its applicability.

The main future directions of this work can be summarized as: (1) the extension of the proposed path planning algorithm to deal with a dynamic environment, in which the target destination and obstacles are dynamic, (2) dealing with non-convex and irregular shapes obstacles, and (3) improve the path planning algorithm by embedding a vision system with a single digital camera or stereo camera. This will aid the navigation processes to detect the surrounding environ-

Fig. 18 The distances between the tracked vehicle and the surrounding obstacles

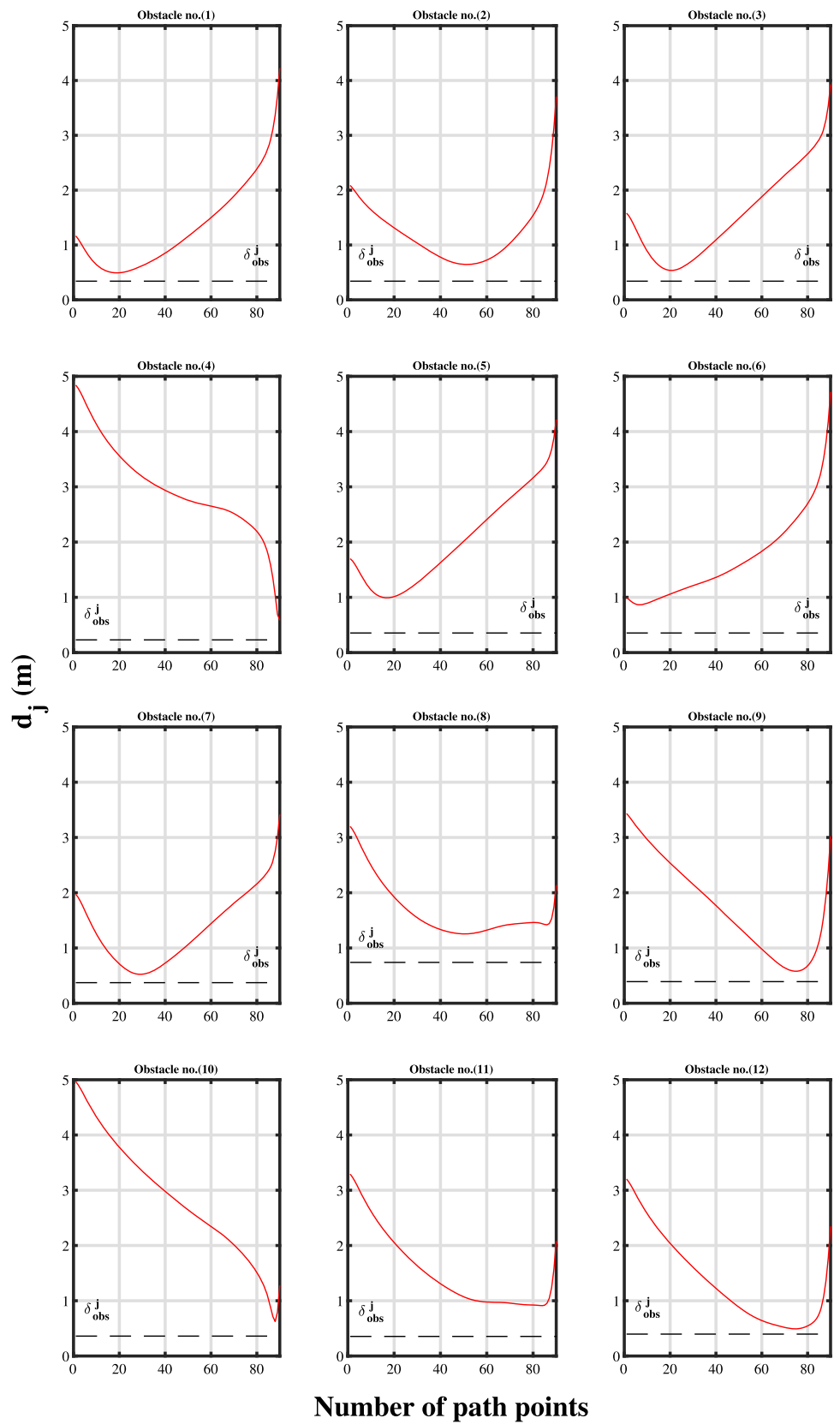
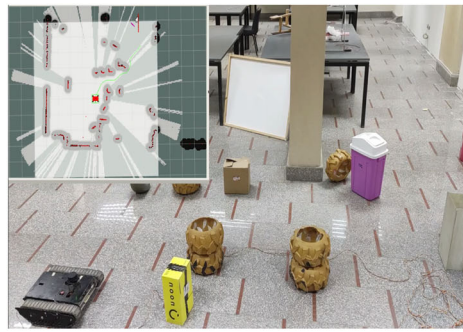


Fig. 19 Real-time snapshots of the tracked vehicle moving between the obstacles through a collision-free path



(a) At time 0 sec (start point)



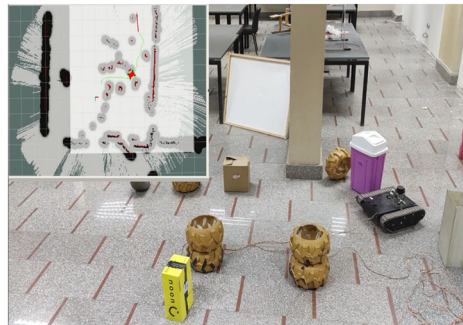
(b) At time 02 sec



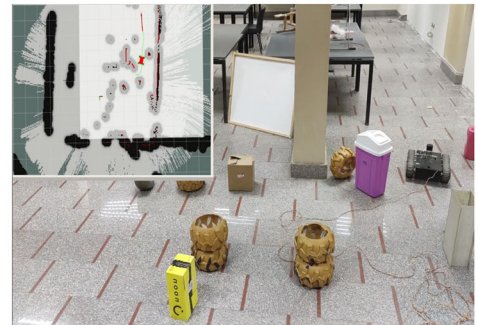
(c) At time 12 sec



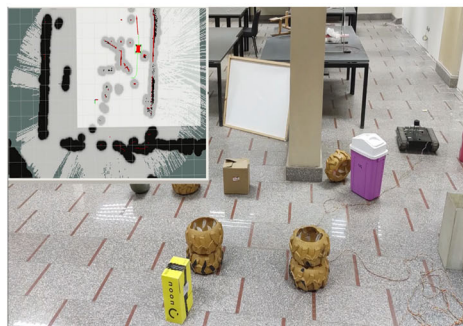
(d) At time 19 sec



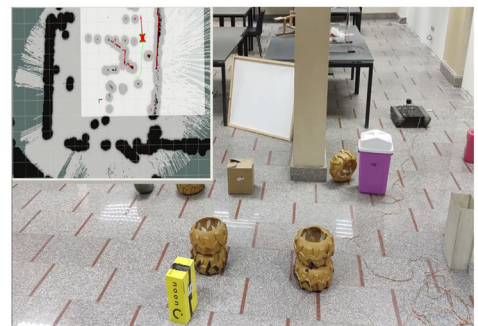
(e) At time 28 sec



(f) At time 37 sec



(g) At time 43 sec



(h) At time 51 sec (end point)

ment as well as the terrain type, results in improving the path planning procedure.

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s11370-022-00429-3>.

Author contribution ADS contributed to conceptual design, methodology, validation, and original draft preparation. MAK contributed to supervision, methodology, validation, thoroughly correction of the manuscript, and final draft preparation. ES contributed to supervision, discussion, and resources. WMH contributed to supervision, discussion, and resources.

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB). No funding was received to do this study.

Data availability All simulation and experimental data in this article are included in this article.

Code Availability Code generated or used in this study can be provided from the corresponding author upon request.

Declarations

Conflict of interest The authors declared that there are no conflicts of interest to declare that are relevant to this study.

Consent to participate Not applicable.

Consent for publication Not applicable.

Ethical approval The authors confirm that there are no potential acts of misconduct in this work, and approve of the journal upholding the integrity of the scientific record.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Raja P, Pugazhenth S (2012) Optimal path planning of mobile robots: A review. *Int J Phys Sci* 7(9):1314–1320. <https://doi.org/10.5897/IJPS11.1745>
- Zhang H, Butzke J, Likhachev M (2012) Combining global and local planning with guarantees on completeness. In: International conference on robotics and automation (ICRA), pp 4500–4506. <https://doi.org/10.1109/ICRA.2012.6225382>
- Patle B, Babu LG, Pandey A, Parhi DRK, Jagadeesh A (2019) A review: On path planning strategies for navigation of mobile robot. *Defence Technol* 15(4):582–606. <https://doi.org/10.1016/j.dt.2019.04.011>
- Injarapu AS, Gawre SK (2017) A survey of autonomous mobile robot path planning approaches. In: International conference on recent innovations in signal processing and embedded systems (RISE), pp 624–628. <https://doi.org/10.1109/RISE.2017.8378228>
- Schwartz JT, Sharir M (1983) On the piano movers problem I. the case of a two-dimensional rigid polygonal body moving amidst polygonal barriers. *Commun Pure Appl Math* 36(3):345–398. <https://doi.org/10.1002/cpa.3160360305>
- Weigl M, Siemiątkowska B, Sikorski KA, Borkowski A (1993) Grid-based mapping for autonomous mobile robot. *Robot Auton Syst* 11(1):13–21. [https://doi.org/10.1016/0921-8890\(93\)90004-V](https://doi.org/10.1016/0921-8890(93)90004-V)
- Choset H, Burdick J (2000) Sensor-based exploration: The hierarchical generalized voronoi graph. *Int J Robot Res* 19(2):96–125. <https://doi.org/10.1177/02783640022066770>
- Choset H, Lynch K, Hutchinson S, Kantor G, Burgard W, Kavraki L, Thrun S (2007) Principles of robot motion: theory, algorithms, and implementation. *Knowl Eng Rev* 22(2):209–211. <https://doi.org/10.1017/S0269888907218016>
- Khatib O (1986) Real-time obstacle avoidance for manipulators and mobile robots. In: Cox JJ, Wilfong GT (eds.) *Autonomous robot vehicles*, pp 396–404. Springer, New York. https://doi.org/10.1007/978-1-4613-8997-2_29
- Sabiha A, Kamel M, Said E, Hussein W (2020) Trajectory generation and tracking control of an autonomous vehicle based on artificial potential field and optimized backstepping controller. In: International conference on electrical engineering (ICEENG), pp 423–428. <https://doi.org/10.1109/ICEENG45378.2020.9171708>. IEEE
- Masehian E, Amin-Naseri M (2004) A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. *J Robot Syst* 21(6):275–300. <https://doi.org/10.1002/rob.20014>
- Cai K, Wang C, Cheng J, De Silva CW, Meng MQ-H (2020) Mobile robot path planning in dynamic environments: A survey. *arXiv preprint arXiv:2006.14195*. <https://doi.org/10.15878/j.cnki.instrumentation.2019.02.010>
- Bhaskar BS, Rauniyar A, Nath R, Muhuri PK (2021) Zone-based path planning of a mobile robot using genetic algorithm. In: Chakrabarti A, Arora M (eds) *Industry 4.0 and advanced manufacturing*. Springer, Singapore, pp 263–275. https://doi.org/10.1007/978-981-15-5689-0_23
- Huq R, Mann GK, Gosine RG (2008) Mobile robot navigation using motor schema and fuzzy context dependent behavior modulation. *Appl Soft Comput* 8(1):422–436. <https://doi.org/10.1016/j.asoc.2007.02.006>
- Kim C, Kim Y, Yi H (2020) Fuzzy analytic hierarchy process-based mobile robot path planning. *Electronics* 9(290):1–18. <https://doi.org/10.3390/electronics9020290>
- Li Q-L, Song Y, Hou Z-G (2015) Neural network based FastSLAM for autonomous robots in unknown environments. *Neurocomputing* 165:99–110. <https://doi.org/10.1016/j.neucom.2014.06.095>
- Sung I, Choi B, Nielsen P (2021) On the training of a neural network for online path planning with offline path planning algorithms. *Int J Inf Manage* 57:1–9. <https://doi.org/10.1016/j.ijinfomgt.2020.102142>
- Tang X-L, Li L-M, Jiang B-J (2014) Mobile robot SLAM method based on multi-agent particle swarm optimized particle filter. *J China Univ Posts Telecommun* 21(6):78–86. [https://doi.org/10.1016/S1005-8885\(14\)60348-4](https://doi.org/10.1016/S1005-8885(14)60348-4)
- Song B, Wang Z, Zou L (2021) An improved PSO algorithm for smooth path planning of mobile robots using continuous high-

- degree bezier curve. *Appl Soft Comput* 100:1–11. <https://doi.org/10.1016/j.asoc.2020.106960>
20. Li F, Fan X, Hou Z (2020) A firefly algorithm with self-adaptive population size for global path planning of mobile robot. *IEEE Access* 8:168951–168964. <https://doi.org/10.1109/ACCESS.2020.3023999>
 21. Xin D, Hua-hua C, Wei-kang G (2005) Neural network and genetic algorithm based global path planning in a static environment. *J Zhejiang Univ-Sci A* 6(6):549–554. <https://doi.org/10.1631/jzus.2005.A0549>
 22. Khelchandra T, Huang J, Debnath S (2014) Path planning of mobile robot with neuro-genetic-fuzzy technique in static environment. *Int J Hybrid Intell Syst* 11(2):71–80. <https://doi.org/10.3233/HIS-130184>
 23. Castillo O, Neyoy H, Soria J, Melin P, Valdez F (2015) A new approach for dynamic fuzzy logic parameter tuning in ant colony optimization and its application in fuzzy control of a mobile robot. *Appl Soft Comput* 28:150–159. <https://doi.org/10.1016/j.asoc.2014.12.002>
 24. Wang X, Shi Y, Ding D, Gu X (2016) Double global optimum genetic algorithm-particle swarm optimization-based welding robot path planning. *Eng Optim* 48(2):299–316. <https://doi.org/10.1080/0305215X.2015.1005084>
 25. Rao RV, Savsani VJ, Vakharia D (2011) Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. *Comput Aided Des* 43(3):303–315. <https://doi.org/10.1016/j.cad.2010.12.015>
 26. Rao RV (2016) Teaching-learning-based optimization algorithm. In: Rao RV (ed) *Teaching learning based optimization algorithm: and its engineering applications*. Springer, Cham, pp 9–39
 27. Savsani P, Jhala RL, Savsani VJ (2014) Comparative study of different metaheuristics for the trajectory planning of a robotic arm. *IEEE Syst J* 10(2):697–708. <https://doi.org/10.1109/JSYST.2014.2342292>
 28. Rao RV, Waghmare G (2015) Design optimization of robot grippers using teaching-learning-based optimization algorithm. *Adv Robot* 29(6):431–447. <https://doi.org/10.1080/01691864.2014.986524>
 29. Wu Z, Fu W, Xue R, Wang W (2016) A novel global path planning method for mobile robots based on teaching-learning-based optimization. *Information* 7(39):1–11. <https://doi.org/10.3390/info7030039>
 30. Aouf A, Boussaid L, Sakly A (2018) TLBO-based adaptive neurofuzzy controller for mobile robot navigation in a strange environment. *Comput Intell Neurosci*. <https://doi.org/10.1155/2018/3145436>
 31. Hernandez-Barragan J (2018) Mobile robot path planning based on conformal geometric algebra and teaching-learning based optimization. *IFAC-PapersOnLine* 51(13):338–343. <https://doi.org/10.1016/j.ifacol.2018.07.301>
 32. Kashyap AK, Pandey A (2020) Optimized path planning for three-wheeled autonomous robot using teaching-learning-based optimization technique. In: Li L, Pratihari DK, Chakrabarty S, Mishra PC (eds) *Advances in materials and manufacturing engineering*. Springer, Singapore, pp 49–57. https://doi.org/10.1007/978-981-15-1307-7_5
 33. Singh G, Sharma N, Sharma H (2020) Shuffled teaching learning-based algorithm for solving robot path planning problem. *Int J Metaheuristic* 7(3):265–283. <https://doi.org/10.1504/IJMHEUR.2020.107391>
 34. Mac TT, Copot C, Tran DT, De Keyser R (2017) A hierarchical global path planning approach for mobile robots based on multi-objective particle swarm optimization. *Appl Soft Comput* 59:68–76. <https://doi.org/10.1016/j.asoc.2017.05.012>
 35. Mohamad SA, Kamel MA (2021) Optimization of cylinder liner macro-scale surface texturing in marine diesel engines based on teaching-learning-based optimization algorithm. *Proc Inst Mech Eng Part J J Eng Tribol* 235(2):329–342. <https://doi.org/10.1177/1350650120911563>
 36. Rao RV, Savsani VJ, Vakharia D (2012) Teaching-learning-based optimization: an optimization method for continuous non-linear large scale problems. *Inf Sci* 183(1):1–15. <https://doi.org/10.1016/j.ins.2011.08.006>
 37. Rao R, Patel V (2012) An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems. *Int J Ind Eng Comput* 3(4):535–560. <https://doi.org/10.5267/j.ijiec.2012.03.007>
 38. Kamel MA, Yu X, Zhang Y (2021) Real-time fault-tolerant formation control of multiple WMRs based on hybrid GA–PSO algorithm. *IEEE Trans Autom Sci Eng* 18(3):1263–1276. <https://doi.org/10.1109/tase.2020.3000507>
 39. Birattari M (2009) *Tuning metaheuristics*. Springer, Berlin. <https://doi.org/10.1007/978-3-642-00483-4>
 40. Wold S, Sjöström M, Eriksson L (2001) PLS-regression: a basic tool of chemometrics. *Chemom Intell Lab Syst* 58(2):109–130. [https://doi.org/10.1016/S0169-7439\(01\)00155-1](https://doi.org/10.1016/S0169-7439(01)00155-1)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.