Engineering Applications of Artificial Intelligence 133 (2024) 108440

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Research paper

In search of the best fitness function for optimum generation of trajectories for Automated Guided Vehicles

Eduardo Bayona [a,b], J. Enrique Sierra-García [a,b,*], Matilde Santos [c], Ioannis Mariolis [d]

[a] Department of Digitalization, University of Burgos, 09001, Burgos, Spain
[b] UBU-MICHELIN Joint Research Unit in Automation and Smart Industry, Arainnov Michelin Aranda, 09400, Aranda de Duero, Spain
[c] Institute of Knowledge Technology, Complutense University of Madrid, 28040, Madrid, Spain
[d] Centre for Research and Technology Hellas, Information Technologies Institute, Thessaloniki, Greece

ARTICLE INFO

Keywords:
Meta-heuristic optimization
Genetic algorithm
Path planning
Fitness function design
Mobile robots
Industrial vehicles

ABSTRACT

This paper presents an offline optimization method designed for use with industrial robots in environments with static obstacles. It is particularly useful in industry where stability and predictability are crucial to meeting expected timelines in automated guided vehicle operations. The main methodological contribution of this work lies in the integral process used to define an effective fitness function that guides the optimization method in the search for optimal solutions. This cost function plays a critical role in the effectiveness of the trajectory tracking algorithm by quantifying path quality and allowing comparisons between solutions. The design of this fitness function poses challenges including accuracy, suitability, minimization of path length, and avoiding or reducing collisions. To achieve the optimization objectives and address some issues such as sensitivity to parameter scaling and the risk of premature convergence, different approaches can be used. This work proposes to incorporate constraints into the fitness function, adjust the optimization parameters to reflect the conditions of the problem, and design a fitness function based on prior knowledge and an accurate representation of the goals. The three relevant contributions for the planning and optimization of routes of automated guided vehicle in industrial environments are the following. Firstly, the development of a mathematical model of trajectories based on Frenet curves that considers the static occupancy map of the environment. Second, an optimization strategy to generate optimal safe paths. Finally, a fitness function that guides the optimization method towards optimal solutions considering the sensitivity to scaling and resolution of the parameters. This study presents an exhaustive analysis of the different fitness functions obtained, each one evaluated based on key metrics such as the length of the trajectory, the average and minimum distance to the occupancy map, and the number of collisions along the path. The results show that the obtained cost function successfully avoids collisions with the environment in all scenarios and consistently remains the fitness function with the largest average distance to obstacles, at least 50% higher than other functions used in this study.

1. Introduction

Today, automated guided vehicles (AGV) are an important asset in the value chain of any industry, whether logistics, manufacturing, distribution, etc. They allow the transport of goods in complex environments (Vakaruk et al., 2021). In the field of robotics and automation, not only the design and mechatronic modelling of the vehicle have become increasingly important but also the optimization of the trajectories followed by AGVs is a key aspect to take into account due to the demands of efficiency and safety in work spaces (Sierra-Garcia and Santos, 2024).

Throughout the process of designing the fitness function that optimizes the path planning, several challenges must be addressed, such as accuracy and feasibility of solutions, path length reduction, and collision minimization. In addition, other issues must be resolved, among which at least two can be highlighted. The sensitivity to the scale of the parameters of the fitness function, which can affect the effectiveness of the algorithm if it is not properly tuned; and the possibility of premature convergence, which can lead the algorithm to suboptimal solutions before adequately exploring the entire search space. This may require including the problem constraints in the fitness function, appropriately tuning the optimization parameters to reflect the problem conditions, and designing a cost function based on prior knowledge and accurate representation. of the objectives of the problem.

* Corresponding author.
E-mail addresses: ebayona@ubu.es (E. Bayona), jesierra@ubu.es (J.E. Sierra-García), msantos@ucm.es (M. Santos), ymariolis@iti.gr (I. Mariolis).

https://doi.org/10.1016/j.engappai.2024.108440
Received 13 December 2023; Received in revised form 28 February 2024; Accepted 8 April 2024
Available online 19 April 2024

Therefore, the purpose of this work is to address the challenges of AGV route planning and optimization in industrial environments. Stability and predictability in trajectory tracking are essential objectives in this context as they affect the ability of AGVs to meet estimated and expected operating times. This work focuses on developing an offline method that can operate efficiently in environments with static obstacles, which are commonly found in industrial facilities. The main methodological contribution is the definition of an effective fitness function that guides an optimization process in the search for optimal solutions for trajectory tracking. This approach addresses various challenges such as accuracy and collision minimization. In summary, this work proposes a comprehensive solution for route planning in industrial environments with AGVs.

The context presented here of fixed obstacles responds to the real world, especially in the industrial environment, where safety is key. The restrictions that come with the tasks carried out by AGV commercial vehicles require guaranteeing a constant cycle time, also called tag time, and perfectly following specific routes planned offline. In these applications there can be no dynamic obstacles to guarantee time and safety. That is why this study is done under these conditions.

Although other research has explored various aspects of route planning, including control system design and AGV route planning, this article provides several novel aspects. It focuses on the development of a mathematical model for trajectories based on Frenet curves, prioritizes the optimization of safe paths, and design a fitness function based on an ad-hoc refinement process to address sensitivity to parameters and avoid premature convergence.

By using this refinement process the final fitness function obtained includes some beneficial features. The absence of coefficients in the fitness function, coupled with the use of continuous parameters, reduces the need to adjust coefficient values based on the optimization scenario, what makes it more adaptive and less sensitive to changes in the optimization scenario. This adaptability is also enhanced by the fact that continuous parameters allow for a smoother and more flexible representation of the optimization problem. Furthermore the piecewise definition and the normalization makes the algorithm more robust since the values of the fitness function does not experience significant fluctuations or discontinuities which facilitates convergence.

The approach has been simulated in scenarios of increasing complexity, in which a series of obstacles have been defined that remain constant over time, with regular geometric shapes such as cubes, parallelepipeds or cylinders, and more irregular shapes, as is common in an industrial environment. The optimization strategy is tested with genetic algorithms (GA) as optimization method, but any meta-heuristic technique could be used. These contributions provide a comprehensive foundation for effective route planning in industrial environments with AGVs.

The main contributions in the planning and optimization of routes for AGVs in industrial environments of this work are:

- The development of a mathematical trajectory model based on Frenet curves that considers the static occupancy map of the environment.
- An optimization strategy to generate optimal safe paths using the distance of the vehicle from obstacles as a variable to achieve the maximum possible separation of them.
- Design methodology of a specific fitness function to guide the optimization method towards optimal solutions, addressing challenges such as sensitivity to parameter scaling, resolution and avoiding premature convergence.

The remaining part of the paper is structured as follows. Section 2 summarizes related works. Section 3 presents the modelling of the problem, including the occupancy map, the trajectory definition, and the collisions with obstacles. Section 4 describes the optimization methodology used to find the optimal solution for the trajectory generation problem. Section 5 presents the results of the iterative refinement process of the fitness function. Results for different scenarios are compared and discussed in Section 6. Finally in Section 7 conclusions and future works are commented.

2. Related works

The robot path planning problem can be categorized into classical and heuristic methods, as shown in Fig. 1. Among the classical methods, remarkable ones include the cell decomposition method (CD), potential field method (PFM), subgoal method (SG), and sampling-based methods. In the cell decomposition method, the robot free configuration space is divided into small regions called cells to provide a collision-free path to the goal, assuming movements only in a limited number of directions (Milos, 2007). With the potential field strategy, obstacles are assigned repulsive forces, and the final destination is given attractive forces, allowing the robot to move towards the goal and away from obstacles (Cosío and Castañeda, 2004). In Sfeir et al. (2011), a new repelling potential formula is presented to avoid conflicts when obstacles are near a target. This method is also extended in Kang et al. (2011) for multi-robot navigation and complex tasks. Candido et al. introduces in Candido et al. (2008) hierarchical subgoal planners for humanoid robot navigation, and Liu et al. (2010b) proposes a real-time approach for this method, dynamically generating subgoals based on time and potential values.

Within the sampling-based planning schemes, Probabilistic Roadmap (PRM) and Rapidly-Exploring Random Trees (RRT) are considered the most commonly used methods (Lee et al., 2014). Yan et al. (2013) presents an innovative approach of multi-robot motion planning using a probabilistic roadmap based on adaptive cross sampling (ACS). This approach involves three steps, including environment space sampling, roadmap construction, and motion planning. Ladd and Kavraki (2004) provides an interesting analysis of PRM for path planning, conceptualizing the problem as the calculation of transitive closure within a probability space. It also sets an upper bound on the anticipated number of PRM iterations required to discover a path.

On the other hand, Rapidly-Exploring Random Trees exhibits high computational efficiency and effectiveness, and the ability to find a feasible motion path relatively quickly, even in high-dimensional spaces. In Corominas Murtra et al. (2010), to avoid obstacles, the authors combine a local planner implemented with RRT with a slightly modified dynamic window approach. To overcome the challenge of generating a branching trajectory in the workspace when using randomization techniques, Ardiyanto and Miura (2012) employs a path planning approach for a mobile robot in dynamic and cluttered environments with kinodynamic constraints.

However, classical approaches do not always yield optimal paths and often get stuck in local minima. Moreover, some of them may not provide suitable solutions in the presence of multiple obstacles or in dynamic environments. To mitigate the inefficiency of classical methods, heuristic approaches are used in this work (Masehian and Sedighizadeh, 2007).

Heuristic methods, as discussed in Mac et al. (2016), can be categorized into neural networks (NN), fuzzy logic (FL), nature-inspired methods (NIM), and hybrid algorithms. Neural networks offer advantages such as nonlinear mapping, learning capability, and parallel processing. For instance, in Yang and Meng (2000), the proposal of using a neural network for robot trajectory planning in a non-static environment is presented. From a different perspective, in Al-Sagban and Dhaouadi (2012) a novel neural network approach that uses a reactive navigation algorithm in unstructured indoor environments is introduced. Their method includes both online and offline learning phases to achieve optimal performance. Fuzzy logic is also applied to represent subjective uncertainties in the human mind, allowing for a set of *if-then* rules to be used for robot navigation decision making. A method for navigating a robot in an unknown and complex

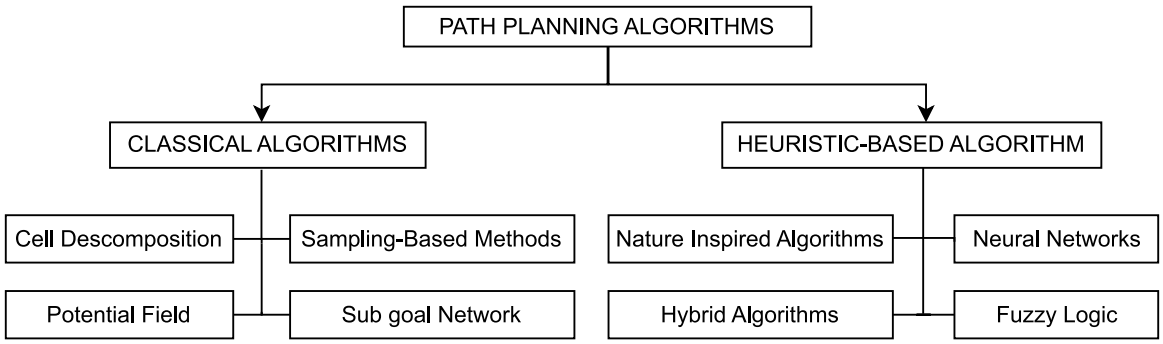


Fig. 1. Path planning algorithms classification diagram.

environment with fuzzy logic and stereo vision-based path planning is presented in Abdessemed et al. (2014). An ordered hierarchical architecture based on fuzzy reasoning is also proposed in Mester (2008) to increase autonomy in cluttered environments. In this work the navigation of an autonomous mobile robot in an unknown environment with obstacles is addressed. The distance and orientation between the robot, the obstacle and goal are taken into consideration. In this paper, the advantages of nature-inspired methods is leveraged to devise a robust and adaptive approach for automated path planning in industrial settings.

Recently, robot navigation techniques inspired by biological behaviours, known as biomimetic algorithms, are gaining more attention. Also in Mac et al. (2016), three methods are highlighted due to their relevance in the literature: genetic algorithms (GA) (Santiago et al., 2017; Chen et al., 2022), particle swarm optimization (PSO) (Han et al., 2019), ant colony optimization (ACO) (Iser and Wahl, 2010). These methods approach trajectory planning as an optimization problem. This strategy involves defining in mathematical terms the objective function, decision variables and constraints. GA is an optimization technique based on natural genetics that takes advantage of mechanisms such as natural selection, crossover and mutation. GA is a powerful tool for solving combinatorial optimization problems and is the approach used in this article, which will be discussed later (Sánchez-Ibáñez et al., 2021).

PSO, similar to GA, generates a random population and evaluates particles using an objective function. However, unlike genetic algorithms, PSO does not include crossover and mutation operations. Initially, the social behaviour of bird flight or group movement of fish inspired the development of PSO. At each generation, PSO is initialized with a set of random solutions and updated according to an optimal scheme. In Zhang et al. (2013), an algorithm for planning trajectories of multiple robots based on PSO in an uncertain environment is presented. The fitness function includes the risk factor and the trajectory distance. In Wang et al. (2009), the trajectory planning for a group of robots is formulated as an optimization problem with obstacle avoidance constraints and V-shaped formation in a dynamic environment. The fitness function is determined by minimizing the group's trajectory while preserving the V-shaped formation.

Finally, similar to the PSO algorithm, ACO is a clustering algorithm that implements group behaviour. ACO is based on the behaviour of ant colonies. Interactive communication between ants is at the heart of this behaviour and enables them to determine the shortest path between their nest and food sources. In Englot and Hover (2011) an algorithm for solving multi-objective planning issues in the presence of obstacles is presented. In this investigation, the ant colony optimization approach is integrated with a path planning algorithm that relies on point-to-point sampling. Another example is (Chen et al., 2011), where a two-stage ACO model is proposed. This model is able to overcome the main problem of inconsistency between premature convergence and the optimal trajectory.

In recent years, the GA-based method of trajectory planning, used in this study, has been receiving considerable attention due to its effectiveness in generating optimal trajectories in high-dimensional spaces (Patle et al., 2019). The research of Bayona et al. (2023b) proposes an optimization method that combines genetic algorithms (GA) with binary grid occupancy maps to produce the shortest and most feasible collision-free paths between three given sites. In this work the fitness function uses the path length and a collision coefficient as parameters. Likewise, authors in Ilin et al. (2022) propose a hybrid GA approach for the travelling salesman problem (TSP) within a reasonable computational time for large problems, which helps to avoid premature convergence of the GA to the local optimum. The authors in Alouache and Wu (2018) use two fitness functions to refine the trajectory tracking of a mobile robot, the first used to optimize a PID controller and the second to estimate the trajectory reference of the hybrid system. All these studies show the effectiveness and flexibility of genetic algorithms in solving various optimization problems.

In this article we present a novel approach to AGV path planning designed for industrial settings which methodology centres around a trajectory mathematical model based on Frenet curves, incorporating the static occupancy map of the environment. In comparison to traditional methods such as cell decomposition used by Milos et al. in Milos (2007), this model provides a more detailed understanding of the industrial landscape. This approach ensures the development of safe and efficient routes for AGVs by incorporating the static occupancy map of the environment.

In addition, an innovative optimization strategy to enhance operational efficiency has been developed. Although existing probabilistic sampling methods such as PRM have been explored by Lee et al. in Lee et al. (2014), the presented strategy is uniquely adapted to the dynamic challenges associated with industrial environments, and by balancing safety and efficiency, our approach generates optimal paths that optimize AGV operations.

Additionally, key challenges, such as parameter sensitivity, have been addressed by designing a tailored fitness function to guide optimization methods. Although hybrid approaches combining genetic algorithms with fuzzy logic have been proposed by the authors in Mac et al. (2016), the refinement of this concept in this paper ensures greater adaptability and robustness in industrial contexts.

The work of Lamini et al. (2018) proposes the use of an improved genetic algorithm (GA) for route planning in a static environment represented by an occupancy grid map. Its fitness function evaluates the path quality in terms of path length and energy utilization. This study will be used for comparison purposes with respect to our work.

An innovative approach to the implementation of the fitness function is carried out by the authors of Liu et al. (2010a). It focuses on continuous optimization-based refinement aimed at adjusting the reference trajectory to avoid obstacles in environments with unknown characteristics. This methodology involves two different planners: global and local. The first is responsible for refining the initial trajectory when encountering or approaching obstacles, and the second calculates

Table 1
Comparison of memory requirements between binary and geometric occupancy map. Size of the map is 10 m × 10 m, obstacles have 4 points (*pointsObs*) and there are 20 obstacles (*nObs*) in the map.

Precision	Binary	Geometric
m. (Metres)	10 × 10 = 100 Bytes	$2 * pointsObs * nObs = 160$ Bytes
mm. (Milimetres)	10000 × 10000 = 100 MBytes	$4 * pointsObs * nObs = 320$ Bytes

an optimal control policy, with this local planner focusing on safety, dynamic feasibility and precise trajectory tracking, especially during manoeuvres at low speed.

As shown, previous research has explored aspects of path planning, including control systems design and challenges faced by AGV path planning. The fitness function is essential for the effectiveness of the algorithm as it measures the quality of generated paths and enables comparisons between different solutions. This paper aims to investigate specific contributions in this field that have not yet been addressed. One of the methodological contributions of this work is a comprehensive process for defining an effective fitness function that guides the optimization method in the search for the optimal solution. Based on the literature review, to the best of our knowledge, no similar analyses have been conducted to those performed in this article.

3. Description and modelling of the problem

This paper develops an optimization strategy to design safe paths for AGVs in industrial environments. To accurately define these trajectories, the mathematical model of Frenet curves is used. Prior to the determination of these trajectories, the static occupancy map upon which the AGV will travel is modelled along with the identification of potential collisions with environmental obstacles.

3.1. Environment modelling

The initial phase of the optimization involves the definition of the obstacles and the corresponding occupancy map on which the vehicle will operate. Precise characterization of the dimensions, shapes and locations of obstacles is important. The trajectory planning phase is reliant on this particular occupancy map as its main input.

Occupancy maps provide information about the workspace and the obstacles within it. Different methods have been proposed to generate occupancy maps, such as maps based on geometric shapes (Wolter et al., 2004), binary occupancy maps (Toda and Kubota, 2011) and signed distance fields (Liu et al., 2021). In contrast to previous papers that use occupancy maps based on grids and straight lines to define routes (Hu and Yang, 2004; Yang et al., 2006), this paper presents an approach that generates an occupancy map using geometric shapes defined by closed polylines. This approach simplifies the calculation of the distance between the vehicle and obstacles, and allows us to exploit it as a trajectory optimization parameter. It leads to more efficient calculations and allows the use of the distance to obstacles as a continuous optimization parameter instead of a discrete one, improving the process through a better definition of the fitness function.

It is advisable to use integers when defining the error in an occupancy map. The primary justification is the consistent accuracy and predictability that integers provide compared to floating point numbers, that can have different degrees of precision. The precision is much better for smaller numbers than for larger ones. This issue could arise when planning a path for industrial robots. Integers provide a suitable representation of the positions of the obstacles in the plane with a fixed and known precision, ensuring that requirements are met. This approach avoids rounding problems and cumulative errors in path calculation. Furthermore, using integers simplifies implementation and reduces computational complexity, which ultimately results in better performance and higher reliability.

When comparing the memory usage of both map types (binary and geometric), it becomes apparent that employing a geometric occupancy map results in a significantly smaller memory usage than the binary case, making it more efficient to manage. In this comparison, it is noted that not only the memory required to define each cell has to be considered, but also the memory required to distinguish between an occupied cell and a free cell. Similarly, in the case of a geometric map, the map's definition relies on the points forming the polyline that delimits the work area.

To provide a straightforward comparison, let us consider a 10 × 10 m environment with *nObs* obstacles present in the map. If the precision is set to metres, the memory usage in the map is 10 × 10 bytes, where obstacles are represented by cells with one occupancy bit. On the other hand, when the precision is set to millimetres, the memory usage in the binary map grows to 10000 × 10000 bytes. In the geometric case there is no fixed quantity of memory required and it will vary in each case, also depending on the number of points that form the polyline that delimits each obstacle. For instance, suppose we deal with obstacles described by four points. Then Table 1 shows the memory usage allocated depending on the particular circumstances of each scenario.

Assuming that each obstacle shape is defined by four points and that there are 10 obstacles in the map, it is observed that the geometric approach may result in greater memory usage to define the map in situations where the accuracy is low. However, as accuracy increases, the memory consumption in the binary case dramatically increases, making the geometric case significantly more efficient from a memory management point of view. It should be noted that when dealing with complex obstacles and numerous instances of them, memory usage will increase proportionally with the geometric approach, but remain constant in the binary approach due to its modelling features.

In the geometric map, the number of bytes required per point depends on the size of the map and the precision. This is the first multiplier in Table 1. For instance, if the size of the map is 10 × 10 m and the precision is 1 m, 1 Byte is enough for each coordinate of each point, as with one byte we can cover a distance of 256 m. Thus 2 bytes are needed per point as we consider x and y coordinates. On the other hand, when the precision is set to millimetres, we need 2 bytes per coordinate, and thus 4 bytes per point. With 2 bytes we can cover a distance of 65536 mm.

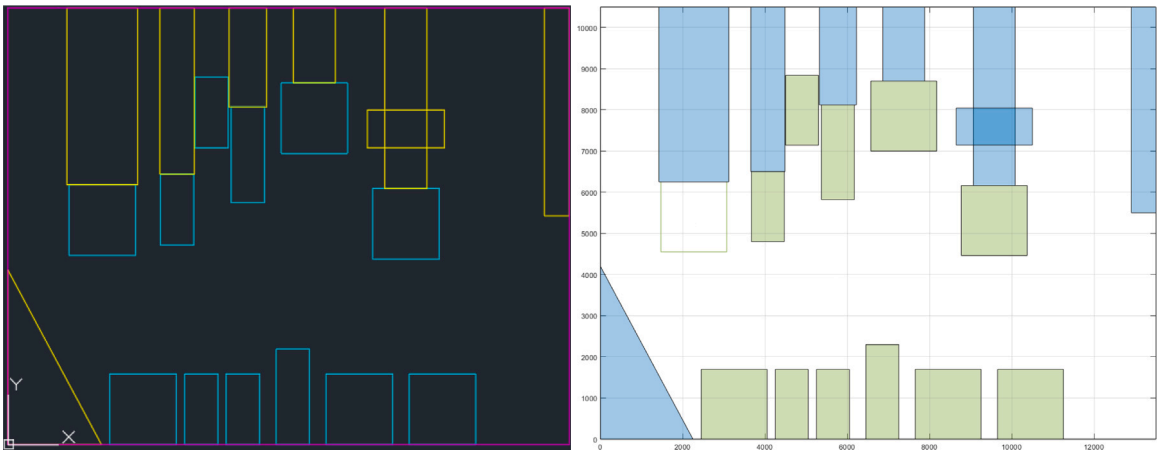
In this paper, we utilize a geometric occupancy map with millimetre precision, which results in significantly lower memory consumption, thus improving efficiency and simplicity of the computations but keeping enough accuracy.

CAD software is used to create the occupancy map using polylines. Different layers define the obstacles and the map's boundaries. The result is shown in Fig. 2(a). Layers can also be used to incorporate annotations in the map such as obstacle types.

For the mathematical modelling of the map, as it is shown in Fig. 2(b), a new LISP script was created to automatically generate obstacles. The occupancy map is then created using geometric shapes to define the spaces inside the obstacles as zones of occupation, leaving as unoccupied the zones between the obstacles and the boundaries of the map. This automated process guarantees a precise representation of obstacles on the map (Bayona et al., 2023a).

3.2. Trajectory modelling

In target tracking, it is crucial to estimate fundamental kinematic quantities, including position and velocity, to know the dynamics of the objects to be tracked. To model this trajectories, differential geometry



(a) CAD model (b) Mathematical model

Fig. 2. CAD and mathematical occupancy map models.

is used, as it allows the calculation of kinematic characteristics of an object following a specific path in space (Kreyszig, 2011). This theory not only describes the object's path, but also specifies important parameters, such as the path's curvature and torsion, which are components of the differential equations that describe the dynamics of three elementary vectors: the tangent, normal and bi-normal vectors, comprising the Frenet–Serret formulae (Martins et al., 2018). These trajectories contain information about the object's position and velocity (Werling et al., 2010).

By using Frenet curves, path planning in complex environments can be more efficient, improving smoothness, obstacle avoidance and adaptability, overcoming the limitations of grid-based approaches, resulting in more natural trajectories. Some previous research has used Frenet–Serret formulas to represent simulated trajectories (Xing et al., 2022). For example, Liu et al. introduce the Frenet–Serret formulas to describe the direction and parametrically express the trajectory of the curve as a function of time and thus allow a CAD system to transfer this information to a CNC machine (Liu et al., 2010a). Kusuma et al. present a trajectory planning method for autonomous vehicles in dynamic road conditions with the ability to avoid moving obstacles using the Frenet reference path model, and then uses a new fitness function to determine the optimal trajectory (Kusuma et al., 2022).

Frenet curves have promising applications in various industries and sectors. They can streamline processes such as CNC machining and welding in manufacturing by ensuring accurate and efficient tool path generation. In autonomous vehicles and robotics, Frenet curves enable smooth and adaptive trajectory planning, facilitating obstacle avoidance and improving navigation in dynamic environments. Additionally, Frenet curves can be used to optimize route planning for delivery vehicles in logistics and supply chain management, resulting in increased efficiency and reduced transportation costs. Their versatility makes them valuable in various fields where precise and adaptive motion planning is essential. For instance, Frenet curves have been used to design roads in urban areas (Amiridis and Psarianos, 2015).

Clothoid curves have various applications, among others the planning of trajectories for autonomous vehicles or robots, as shown in a study by Dai and Cochran (2009). A clothoid is a curve whose curvature k increases linearly with the length of its arc. The smooth variation of curvature allows seamless transitions between straight segments and curves, this feature avoids steps in the centrifugal forces making them ideal for establishing feasible routes (van der Molen, 1992). The curvature at a given arc length s is computed by the Eq. (1).

$$k(s) = \frac{s}{A^2} \tag{1}$$

In this work, an algorithm, following Bertolazzi and Frego (2015), is applied to solve the Hermite G1 interpolation issue by using a clothoid

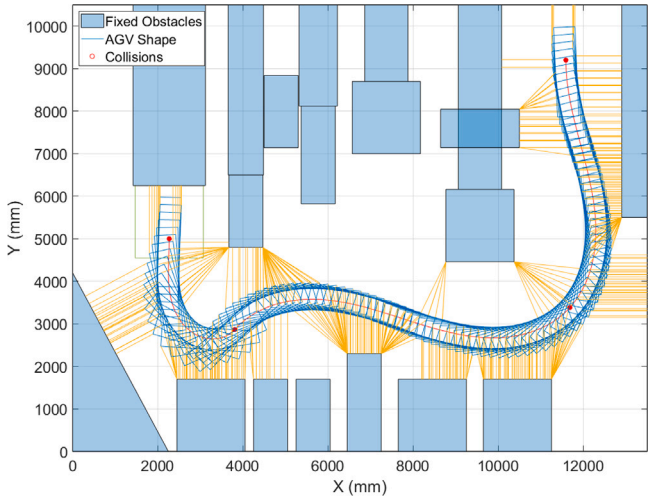


Fig. 3. Successful optimized trajectory in the geometric occupancy map.

curve between two points with unit tangent vectors in a plane. The interpolation problem is expressed as a set of three nonlinear equations, which have a number of possible solutions.

Fig. 3 shows an example of a successful trajectory generation. It is displayed in red, where the red dots represent the input parameters (starting point, ending point, and intermediate waypoints). The AGV at each point of the trajectory is shown in blue. Additionally, yellow lines indicate the minimum distance between the vehicle's vertices and the obstacles in the occupancy map.

To implement this solution, the Matlab function 'referencePathFrenet' is used. This function fits a smooth, continuous, step-wise curve to a set of reference points provided. The curve is a Clothoid and it is fitted by the algorithm defined in Bertolazzi and Frego (2015). The trajectory is then defined in Frenet coordinates, which makes path planning and control easier for autonomous vehicles. Defining a reference trajectory in this coordinate frame enables users to generate detailed trajectories that consider the changes in curvature and other kinematic aspects along the path, which is essential for accurate navigation of AGVs and other autonomous vehicles.

The trajectory's points are expressed via Frenet's equations, which define a two-dimensional curve in $\mathbb{R}^2$ space as $[x, y, \theta, K, dK, s]$, where x, y represent the coordinates of each point in mm, θ denotes the output angle in radians, K, represents the curvature or the inverse of the radius, in m^{-1}, dK is the curvature derivative regarding the arc length

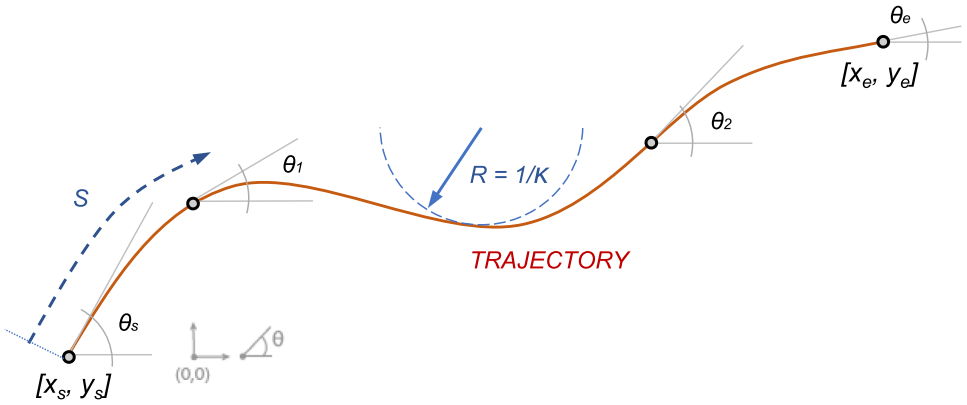


Fig. 4. Trajectory parameters graphical representation.

in m^{-2}, and s stands for the arc length or distance along the path from the path origin in mm.

As initial parameters for the trajectory generation, the starting point (x_s, y_s), ending point (x_e, y_e) and intermediate waypoints (x_i, y_i) are first selected and their respective output angles (θ_i) are the solution outputs of the iterative optimization process. All these parameters and an example representation of a trajectory is shown in Fig. 4.

The generated trajectory is represented by the Frenet equations, which define a flat curve in $\mathbb{R}^2$. The tangent vector T and the normal vector N satisfy Eq. (2), known as the Frenet equations of a trajectory (Alencar et al., 2022).

$$\begin{aligned} T'(s) &= K(s)N(s) \\ N'(s) &= -K(s)T(s) \end{aligned} \tag{2}$$

where s is the distance from the origin at each point, K is the curvature of the trajectory at distance s, $T(s)$ is the tangent vector at distance s, $N(s)$ is the normal vector at distance s, and the symbol $'$ denotes the perpendicular vector.

The arc length of the path can be calculated in Cartesian coordinates. The path is a plane curve in R2 with the form $y = f(x)$, where f is continuously differentiable.

Eq. (3) expresses the length of every infinitesimal segment of the curve.

$$ds = \sqrt{dx^2 + dy^2} = \sqrt{1 + \left(\frac{dy}{dx}\right)^2}\, dx \tag{3}$$

This leads to the form of the arc length s of the calculated trajectory, based on (3), which is expressed by Eq. (4):

$$s = \int_{(x_s, y_s)}^{(x_e, y_e)} \sqrt{1 + \left(\frac{dy}{dx}\right)^2}\, dx \tag{4}$$

where (x_s, y_s) and (x_e, y_e) are the selected start and end points of the trajectory.

3.3. Collision detection modelling

In this study, the distance between the vehicle and obstacles in the occupancy map is used for collision detection. Distances relative to the vehicle are obtained using its vertices as reference points. The position of each vertex of the vehicle is determined based on the dimensions of the vehicle, considering that the centre of the vehicle follows the trajectory under evaluation. This trajectory is segmented into N equidistant samples, generating points along the path where the distances from the AGV vertices to the obstacles will be evaluated. These locations not only determine the potential positions of the vehicle along the trajectory but also establish the positions of the vehicle vertices. Distances from each vertex to surrounding obstacles are calculated for each vertex at each sampled point along the trajectory.

Specifically, to determine the distance between the obstacles and each vertex of the AGV at the points sampled along the trajectory, the projection of each vertex on the straight line that contains each obstacle segment is obtained. The projection of a point P on the line containing a segment $\overline{AB}$ is obtained by (5).

$$proj(P, \overline{AB}) = \frac{\vec{AB} \cdot \vec{AP}}{\|\vec{AB}\|^2} \cdot \vec{AB} \tag{5}$$

If the projection of the vertex falls within the obstacle segment, the distance is calculated as the norm of the vector connecting the vertex point to its projection. However, if the projection falls outside the obstacle on the line of the segment, the norm of the vector connecting the vehicle vertex point to the nearest vertex of the obstacle segment is used. Fig. 5 shows an example of these two cases. In the first case the projection of the point P_1 onto the straight line containing the segment $\overline{AB}$ (denoted by Q_1) falls out of the segment $\overline{AB}$, and A is the closest point to P_1 of the segment $\overline{AB}$. Thus, the distance to the segment is calculated as the norm $|\vec{AP_1}|$. In the second case the projection of the point P_2 onto the straight line containing the segment $\overline{AB}$ (denoted by Q_2) falls into the segment $\overline{AB}$. Thus, the distance to the segment is obtained as the norm $|\vec{Q_2P_2}|$.

If the projection of the vertex falls within the obstacle segment, the distance is calculated as the norm of the vector connecting the vertex point to its projection. However, if the projection falls outside the obstacle on the line through the segment, the norm of the vector connecting the vehicle vertex point to the nearest vertex of the obstacle is used. Fig. 5 shows an example of these two cases. In the first case the projection of the point P_1 onto the straight line containing the segment $\vec{AB}$ (denoted by Q_1) falls out of the segment $\vec{AB}$ and A is the closest point to P_1 of the segment $\vec{AB}$. Thus, the distance to the segment is computed as the norm $|\vec{AP_1}|$. In the second case the projection of the point P_2 onto the straight line containing the segment $\vec{AB}$ (denoted by Q_2) falls into the segment $\vec{AB}$. Thus, the distance to the segment is computed as the norm $|\vec{Q_2P_2}|$.

The method for calculating the distance between a point P and a segment $\overline{AB}$ described in Fig. 5 can be formalized by expression (6), where the distance between any point P and any segment $\overline{AB}$ is denoted by the function $dis_{PS}(P, \overline{AB})$, and $dis_{PP}(A, B)$ is the Euclidean distance between any points A and B, given by Eq. (7), i.e., the norm of the vector connecting points A and B.

$$dis_{PS}(P, \overline{AB}) = \begin{cases} dis_{PP}(P, proj(P, \overline{AB})) & \text{if } proj(P, \overline{AB}) \in \overline{AB} \\ min\left[dis_{PP}(P, A), dis_{PP}(P, B)\right] & \text{if } proj(P, \overline{AB}) \notin \overline{AB} \end{cases} \tag{6}$$

$$dis_{PP}(A, B) = \sqrt{(A_x - B_x)^2 + (A_y - B_y)^2} \tag{7}$$

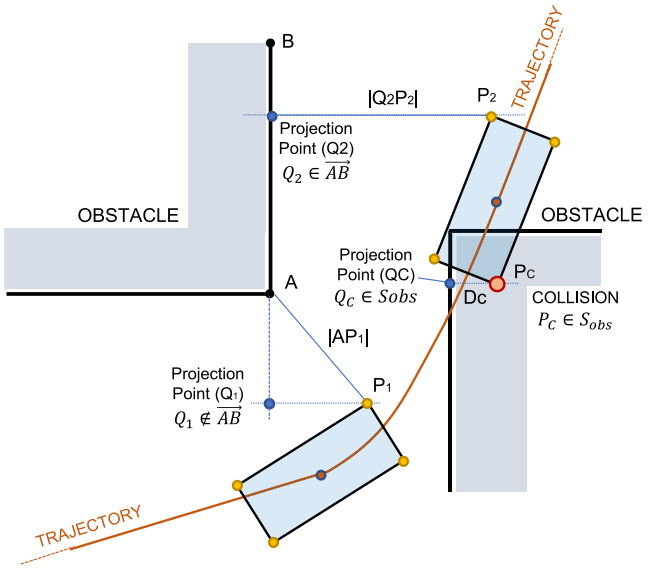


Fig. 5. Visualization that shows distances from trajectory points to obstacles, distinguishing between points without collision and points of collision with obstacles.

Using these definitions, the distance from a point P to an obstacle O can be expressed as the minimum value between the distances from point P to each of the segments that form the obstacle O. This is formally expressed in Eq. (8), where $dis_{PO}(P,O)$ is the distance between point P and obstacle O.

$$dis_{PO}(P,O) = min_{\overline{AB}\in O}\ dis_{PS}(P,\overline{AB}) \tag{8}$$

In this work, collisions are identified by analysing whether the point of the AGV from which the distance is calculated is within the area of the obstacle, in which case it will be a collision of the vehicle with the environment. The distances, both collision and non-collision, to each of the obstacles on the occupancy map are calculated along the vertices of the vehicle at each point of the trajectory. This is formalized by Eq. (9), where D_{ijk} and $D_{c_{ijk}}$ represent the non-collision and collision distances, respectively. The distance from point P_{ik} to obstacle O_j is $dis_{PO}(P_{ik},O_j)$, and S_{O_j} is the area of the obstacle O_j. N is the number of points on the trajectory, N_o is the number of obstacles. P_{ik} denotes the location of the vertex k of the vehicle when the vehicle is at point i of the trajectory. Thus, D_{ijk} represents the distance between the point P_{ik} to the obstacle O_j when there is not collision, and $D_{c_{ijk}}$ when there is collision.

$$[D_{ijk}, D_{c_{ijk}}] = \begin{cases} [0, dis_{PO}(P_{ik},O_j)] & \text{if } P_{ik} \in S_{O_j} \\ [dis_{PO}(P_{ik},O_j), 0] & \text{if } P_{ik} \notin S_{O_j} \end{cases}, i \in 1..N, j \in 1..N_o, k \in 1..4 \tag{9}$$

For each position i, of the vehicle, the distances from its vertices to each obstacle j are evaluated, and the lowest of these values is chosen to define the minimum distance from the vehicle to the occupancy map at this particular point in the trajectory. This is formalized in Eq. (10) for the case of no collision, and with (11) for the case of collision of the vehicle with the environment.

$$Dc_i = min_{k=1...4, j\in 1..N_o}(D_{c_{ijk}}), i \in 1..N \tag{10}$$

$$D_i = min_{k=1...4, j\in 1..N_o}(D_{ijk}), i \in 1..N \tag{11}$$

Finally, based on this, the total collision distance is defined as the sum of all individual collision distances, as formalized in (12).

$$D_c = \sum_{i=1}^{N} D_{c_i} \tag{12}$$

4. Optimization methodology

To automate the design of trajectories for AGVs, various methodologies can be used. This work provides a comprehensive overview of the complete procedure proposed to design optimized trajectories, as shown in Fig. 6. The occupancy map is generated using CAD software and polylines are used to delimit obstacles during the operation of the AGVs and the boundaries of the map. A LISP script is used to select all objects that make up the occupancy map. Each point of the polylines of the map's objects is then exported to a text file with their corresponding properties, including the identity code of the object, the layer it belongs to, and its coordinates within the limits of the occupancy map. These points are then imported into the mathematical tool based on their exported properties. The occupancy map is created using the points and geometric shapes defined in the CAD tool to represent both the map boundaries and the obstacles. Any space within the geometric shapes that describe the points of each obstacle is considered an occupied zone, while the space between the geometric shapes and the limits of the map is considered free space.

Genetic algorithms are particularly effective in solving complex nonlinear optimization problems with high-dimensional search spaces. This makes them ideal for trajectory optimization tasks in environments with numerous constraints and variables. Its population-based search strategy allows multiple potential solutions to be explored simultaneously, leading to efficient navigation through the vast solution space and identification of high-quality trajectories. Its stochastic nature and diversity maintenance mechanisms prevent premature convergence towards suboptimal solutions, ensuring continuous exploration of promising regions of the solution space as conditions evolve.

To determine the best path for the AGV, an iterative computational process shown in Fig. 7 is proposed. This approach involves refining the trajectory by iteratively adjusting the output angles of the intermediate waypoints θ_i until an optimal solution is reached. Although various metaheuristic optimization techniques could be considered for this task, genetic algorithms (GA) has been selected.

The user defines the starting point, ending point, and intermediate points of the trajectory as initial input for the optimization methodology. This technique provides the output angles of the intermediate waypoints (θ_i), which are crucial for determining the trajectory. These optimized output angles together with the waypoint coordinates (x_i, y_i), and starting and end points coordinates and angles (x_s, y_s, θ_s), and (x_e, y_e, θ_e) are used to generate the trajectory as explained in Section 3.2. The trajectory is tested within the occupancy map to evaluate the distances to the obstacles and the collision checking. This information is considered by the fitness function f to evaluate the performance of the solution. Then the GA optimization receives the value of the fitness function to generate a new set of angles, θ_i.

The parameters of this optimization problem, summarized below, were optimized to provide a balance between computational efficiency and solution effectiveness, tuned through iterative experimentation for the genetic algorithm and simulation environment.

- Constraints and fixed parameters:
  - **a.** Coordinates of the start point: (x_s, y_s)
  - **b.** Output angle of the start point: θ_s
  - **c.** Coordinates of the end point: (x_e, y_e)
  - **d.** Input angle of the end point: θ_e
  - **e.** Coordinates of the waypoints: $(x_i, y_i) \in \mathbb{R}^2\ i \in \{\mathbb{N} \le N_w\}$
  - **f.** Physical dimensions of the AGV

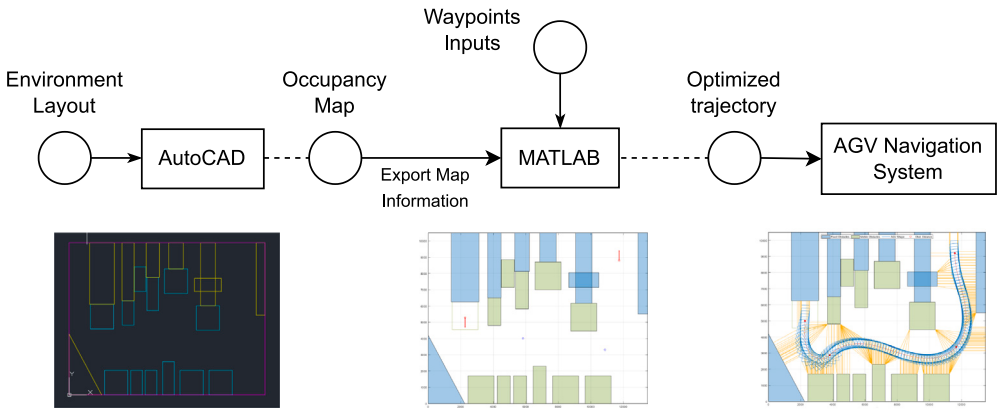


Fig. 6. Flowchart of the process of trajectory optimization.

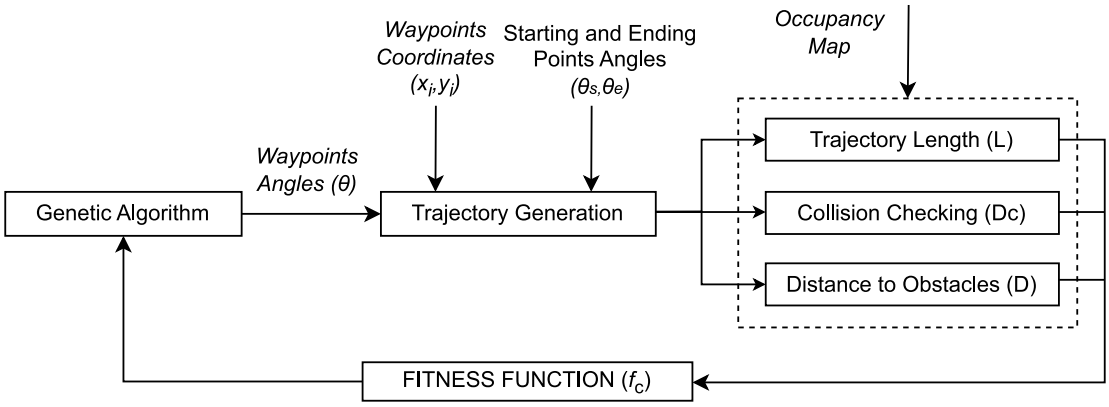


Fig. 7. Flowchart of the complete process of trajectory optimization.

- Optimization variables:

  g. Output angles of the waypoints: $\theta_i \in [0, 2\pi]\ i \in \{\mathbb{N} \leq N_w\}$

Subsequently, considering the vehicle's dimensions and the position of the obstacles in the scenario, the trajectories are evaluate according to possible collisions. Each solution found by the algorithm is then weighted by its fitness function, considering the presence or absence of collisions, in order to find the best solution for each iteration and the optimal solution at the end of the process.

Conclusions drawn from an ablation study highlight the need for each component of the proposed architecture to achieve optimal performance of the method. In particular, it has been shown that both the generation of trajectories and the calculation of collisions and distances are essential elements to achieve satisfactory results in the optimization of AGV trajectories.

4.1. Design process of the fitness function

This study explores the optimization of path planning for Automated Guided Vehicles (AGVs) in industrial settings. Due to the complexity of this task, with varying goals and constraints, customized approaches are required. To do it, a systematic iterative refinement process of the fitness function is proposed in Fig. 8.

The iterative process begins with the development of an initial fitness function tailored to specific requirements, drawing on insights from existing literature and domain expertise. Subsequent phases involve an examination of the theoretical basis and practical implications of the proposed fitness function, with the aim of identifying potential areas for improvement and refinement. The performance of the fitness function was evaluated through simulation experiments conducted in an industrial environment, using diverse scenarios representative of industrial applications. This enabled the identification of any challenges or limitations encountered in its functionality. Based on these findings, a variety of potential solutions are systematically suggested and integrated into successive iterations of the fitness function. Each iteration is designed to improve its adaptability, robustness and overall effectiveness.

The final fitness function obtained following the process in Fig. 8 deals with collision checking as a continuous parameter, rather than as a binary constraint. In this way, a more precise assessment of the collision risk is achieved at each iteration of the optimization process. This approach allows the algorithm to discern subtle variations in collision risk and prioritize routes that minimize this risk, while optimizing other objectives, resulting in a more accurate assessment of the quality of the solution. This continuous nature of the collision avoidance parameter allows for smoother transitions between solutions, improving the algorithm's ability to balance collision risk, path length, and other objectives effectively. The optimization process is more robust and effective, ultimately leading to better routes in a variety of scenarios.

The procedure shown in Fig. 8 explains why different fitness functions are obtained. Each one improves the previous one in some aspect. For this, iterative tests have been carried out and each proposal aims to improve some factor of the solution, such as the length of the path, the proximity to obstacles and the frequency of collisions. The final

Table 2
Different approaches of the fitness function design process.

Collision-based approach	
MLPC: Minimum possible length penalized by collision	$f_c = s \cdot C_c$
MLMC: Minimum length and minimum collisions	$f_c = s \cdot C_c \cdot N_c$
Approach based on distance to obstacles	
MDOBS: Minimum distance to obstacles from the nearest point	$f_c = \frac{1}{min_{j \in N_o}\{D_{ij}, \quad i=1,2,\ldots,N\}}$
AVGMD: Average minimum distance to obstacles	$f_c = \frac{N}{\sum_{i=1}^{N} min_{j \in N_o} D_{ij}}$
PWAVG: Average minimum distance to obstacles considering collision	$f_c = \begin{cases} 1 + \sum_{i=1}^{N} min_{j \in N_o} D_{c_{ij}} & \text{if } D_c \neq 0 \\ 1 - \frac{\sum_{i=1}^{N} min_{j \in N_o} D_{ij}/N}{max_{j \in N_o} D_{ij}} & \text{if } D_c = 0 \end{cases}$

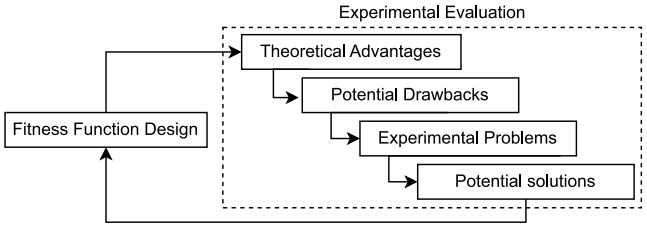


Fig. 8. Flowchart of the fitness function evaluation process.

cost function is considered the best and its best performance has been shown in simulation of the AGV trajectories.

This work provides a comprehensive analysis of the fitness function design process, beginning with an initial search for the shortest collision-free path and concluding with a final approach that considers the average minimum obstacle distance in such scenarios. Table 2 offers a summary of the fitness functions evaluated. These fitness functions have been obtained following the process described in Fig. 8.

In Table 2, s is the trajectory length calculated by Eq. (4); C_c is the collision coefficient, being 1 when no collision is detected and 10^3 when there is collision; n is the number of trajectory points, N_o is the number of obstacles in the occupancy map, D_{ij} denotes the distance from the point i to the obstacle j when there is no collision, and $D_{c_{ij}}$ when a collision occurs. D_c is the sum of distances to all colliding obstacles. Thus, when there is no collision ($D_c = 0$), the fitness function aims to maximize the distance to obstacles. On the other hand, when collisions occur ($D_c \neq 0$), the fitness function is designed to minimize the number of them.

The strengths, weaknesses, and outcomes of every approach will be systematically and thoroughly discussed in Section 5.

4.2. Genetic algorithm configuration

To identify the best solution for any given scenario, as said a genetic algorithm is used to explore the possible output angles of the waypoints. The initial output angles are randomly generated subject to the constraints given in constraint **g**. The *size of the population* has been set to 50.

The *intermediate* crossover operator is used. It randomly selects two parents from the population and averages the corresponding variables of the parents to generate a new spring. The averaging process incorporates a random weight for each variable, increasing the diversity of the offspring.

The *limit* mutation approach is adopted. This operator introduces random modifications to the chosen variables of a solution within a range, in this case 20% of the total variable range.

Individuals are then selected to form the new population through a combination of fitness-proportional selection and elitism. The probability of selection for each individual is determined by normalizing its fitness values. In every generation, two elite individuals, representing the best solutions of the current generation, are directly included in the next generation.

This parameters have been optimized based on a combination of field experience, practical considerations and iterative experimentation. Numerous experiments were performed to explore different parameter settings and evaluate their impact on the performance of the algorithm, with the aim of achieving computational efficiency and a good solution.

5. Iterative refinement of the fitness function

The refinement process of the fitness function follows the structure presented in Fig. 8. This process is carried out until a fitness function that adequately meets the proposed requirements is found. The advantages and potential drawbacks of the fitness function are studied. This analysis is based on experimental results and a new fitness function is then proposed accordingly.

The study begins with an initial fitness function that highlights the problems associated with the experimental setup. Subsequently, a series of refined fitness functions are described, each designed to address the deficiencies of its predecessor. This process helps to improve the understanding of the performance of the optimization process in terms of accuracy, efficiency, and reliability.

5.1. MLPC. Minimum length penalized by collision

The MLPC fitness function is formulated to achieve the shortest possible path by penalizing the presence of collisions with a collision coefficient, as shown in Eq. (13).

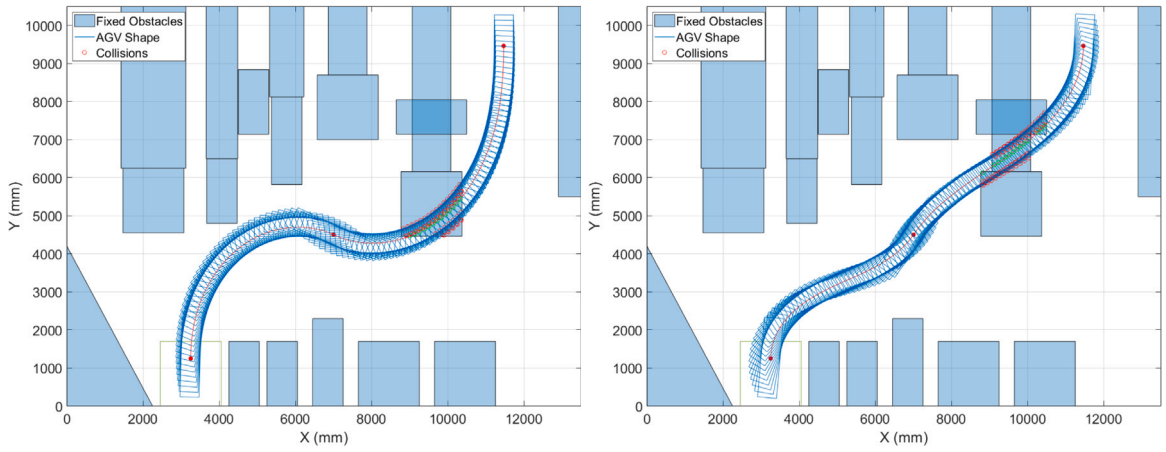
$$f_c = s \cdot C_c \tag{13}$$

where s represents the length of the trajectory obtained by (4), and C_c stands for the collision coefficient, being 1 when no collision is detected and 10^3 when there is collision.

5.1.1. Potential drawbacks

Scale sensitivity could be an issue as the magnitude of the collision coefficient can impact the algorithm's sensitivity. Minor adjustments in the position of obstacles could result in significant changes in the fitness function, potentially hindering the genetic algorithm's ability to converge.

The optimization process may become convoluted due to the length-based and collision-based components of the fitness function, leading to complex search spaces and ultimately suboptimal outcomes.



(a) Solution 1 (b) Solution 2

Fig. 9. Different scenarios results for minimum possible length penalized by collision fitness function.

Determining the value of the collision coefficient may pose difficulties. Selecting a high value may overemphasize penalties and result in suboptimal solutions. Conversely, a low value may not adequately improve collision prevention. Furthermore, if a very high collision coefficient is used, the algorithm may find a solution prematurely because all parameters have very high values due to collisions, so it selects the shortest one that collides. This phenomenon can be named Collision dominance.

The current fitness function fails to distinguish between varying levels of collisions, resulting in equal penalties for both minor and severe collisions. This can result in instances where the fitness function is unable to distinguish between significant risk situations and minor collisions, potentially leading to suboptimal decisions. To address this, the importance of the collision should be quantified based on the distance from the collision to the obstacle edge.

Total number of collisions should also be considered. This type of fitness function fails to consider the quantity of potential collisions that occur throughout the trajectory. This parameter is significant when determining whether the trajectory is undergoing collision-free optimization.

5.1.2. Experimental problems

Fig. 9 shows two solutions provided by the algorithm when this fitness function is evaluated under the same scenario. The value of the fitness function for each solution is:

- Solution 1 $= 1.3898 \times 10^7$
- Solution 2 $= 1.2095 \times 10^7$

Although the fitness function of solution 2 is smaller than for solution 1, it is clear that the solution 2 is worse as shown in Fig. 9(b). Thus, the value of this fitness function is not a proper indicator of the best solutions. In Fig. 9(b), the method shows a behavioural pattern focused on improving the values that define the fitness function, i.e., to decrease the path length, which directly lessens the number of collisions. Nevertheless, the fitness function has a limited capacity to distinguish between various collision contexts, therefore emphasizes solutions in which the path length is notably reduced.

In this scenario, reducing the path length leads to an improvement of the fitness function value. Nevertheless, collision presence continuously dominates and reduces the fitness value as it advances through generations of individuals, implying that reducing the path length will ultimately reach the optimal solution.

5.1.3. Potential solutions

Adjustments to the fitness function are necessary to address the outlined drawbacks. Such adjustments should concentrate on attaining a more precise identification of collision situations related to the trajectory to allow better decision-making.

To achieve this, it is necessary to include the number of collision points (collisions between the vehicle and the occupancy map) in the fitness function. Two approaches can be taken to determine collisions on a trajectory: counting the number of obstacles it collides with, or calculating the number of points on the trajectory that are in collision. The second approach is preferred, as the fitness function achieves smaller values when the trajectory has fewer points in the obstacle map.

5.2. MLMC. Minimum length and minimum collisions

The aim of the MLMC fitness function is to find the shortest possible path by penalizing the presence of collisions not only with a collision coefficient but also by adding the number of collision events, as shown in Eq. (14).

$$f_c = s \cdot C_c \cdot N_c \tag{14}$$

Here, s denotes the length of the trajectory obtained by (4), where C_c stands for the collision coefficient, being 1 when no collision is detected and 10^3 in other case, and N_c is the number of collisions.

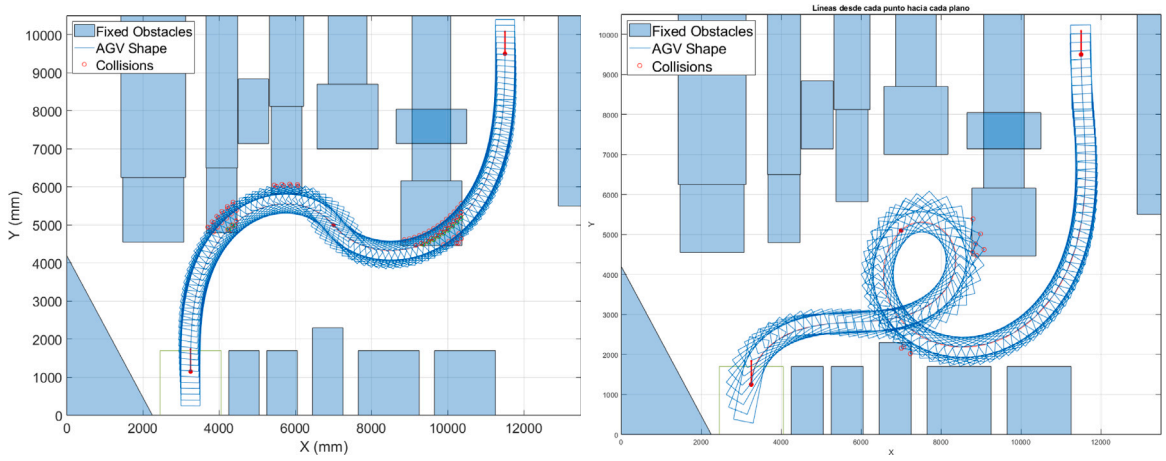
5.2.1. Theoretical advantages

Including collision count in the fitness function directly addresses the parameter scaling sensitivity problem by providing a more holistic metric to evaluate algorithm performance and improve the trade-off between path length and safety, i.e. collision prevention. Without this consideration, the fitness function was based solely on the path length and multiplied by a coefficient in case of collisions, which could be affected by changes in the parameter scale of the function.

By examining the quantity of collision points along the trajectory, it is possible to distinguish between minor and more severe collisions. This may result in a fitness function that more accurately penalizes collisions that represent higher impact with obstacles, that is, further into the occupancy map.

5.2.2. Potential drawbacks

It is noteworthy to consider the sensitivity of the fitness function to the number of points of the trajectory. The trajectory resolution may have an impact on the consistency of the results if different trajectory resolutions are compared.



(a) Solution 1 (b) Solution 2

Fig. 10. Experiment results for minimum possible length with as few collisions as possible fitness function.

When binary maps are used, the calculation of N_c is subjected to uncertainty due to the discretization of the occupancy map and the resolution of the points. This uncertainty can compromise the fitness function's accuracy, which can lead to inconsistent results. In this work this problem cannot appear as we are using geometric maps (Section 3.1).

Furthermore, implementing the fitness function may cause problems due to the lack of smooth convergence gradients in relation to the number of collisions. Depending on the specific scenario or application, challenges may arise, such as discontinuity or lack of differentiability, which can make the optimization process challenging. This is especially important when the optimization methods require smooth convergence.

5.2.3. Experimental problems

Fig. 10 shows two solutions obtained when this fitness function is evaluated under the same scenario. The value of the fitness function for each solution is:

- Solution 1 $= 1.1966 \times 10^9$
- Solution 2 $= 3.6835 \times 10^8$

Although the value of the fitness function for the solution 2 is smaller than for solution 1, it is clear that solution 2 is worse (Fig. 10(b)). So the value of this fitness function is not the best indicator of the optimal solution. On the other hand, this new fitness function appears to be an improvement in terms of more detailed consideration of collisions and the number of collisions.

In Fig. 10(b) it is also possible to see how this fitness function prioritizes the reduction of collisions vs. the path length. However, it is essential to balance those two factors.

In fact, if the weight given to collisions becomes more relevant than the path length, solutions may reduce number of collisions but are not suitable for the precise movement of an AGV, as shown in Fig. 10(b). This can lead the optimization process in an undesirable direction. The results show a high dependency and sensitivity to both, path and vehicle sampling. These parameters largely determine the number of collisions and consequently, influence the direction the process will take in its search for the optimal solution.

5.2.4. Potential solutions

It was decided to abandon path length optimization in favour of a safety-oriented approach, where obstacle distances are used to ensure that the trajectory avoids collisions. We emphasize the role of the fitness function in achieving a trajectory that stays as far away from obstacles as possible.

5.3. MDOBS. Minimum distance to obstacles from the nearest point

The MDOBS fitness function aims to maximize the minimum distance between the vehicle's trajectory and obstacles in the environment. This is expressed by Eq. (15).

$$f_c = \frac{1}{min_{j \in N_o} \{D_{ij}, \quad i = 1, 2, \ldots, N\}} \tag{15}$$

Being N_o the number of obstacles in the occupancy map, N the number of trajectory points and D_{ij} the distance from the point i to the obstacle j when there is not collision.

5.3.1. Theoretical advantages

Firstly, the method takes into account the actual proximity of the trajectory to obstacles by directly considering the minimized distance to these obstacles. This approach overcomes the issues of oversimplification seen in previous fitness functions.

Unlike the previous fitness function, based solely on path length and collision occurrences, the new approach prioritizes safety by maximizing the minimum distance between the vehicle and obstacles along the path. This change reduces the sensitivity of the algorithm to parameter scaling because it evaluates security-related metrics directly, without using coefficients or scaling factors. Furthermore, by optimizing the distance to obstacles, the algorithm remains effective in different scenarios and environments.

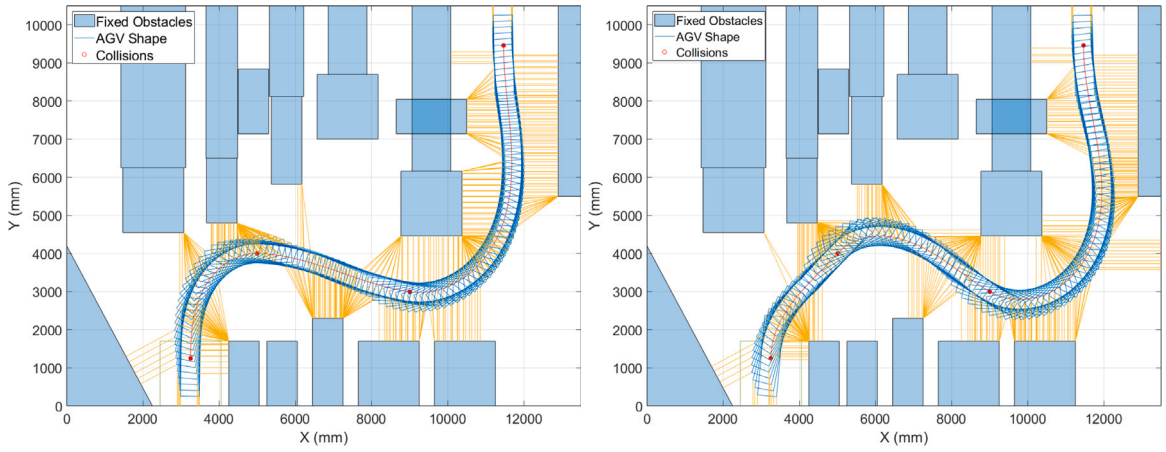
With regards to computational efficiency, determining the minimum distance on a continuous occupancy map is less computationally demanding than on a binary map. This is particularly valid when dealing with coordinate values that avoid using occupancy grids. This approach reduces the number of operations, resulting in efficiency gains overall.

5.3.2. Potential drawbacks

The function may not precisely represent complex or non-uniformly formed obstacles.

In addition, finding the smallest obstacle distance for every point on the route can be computational demanding, particularly when using a discrete occupancy map, as the algorithm must examine numerous map occupancy cells. These computational requirement might have an impact on the efficiency of the algorithm. In the case of binary maps, implementing an exceedingly low resolution may lead to neglecting vital obstacle details, whilst an excessively high resolution may raise computational complexity.

Moreover, the efficiency of the discrete occupancy map could decrease when handling moving obstacles, since the probable occupancy values might not be adequate when managing distinct layers that



(a) Scenario 1 (b) Scenario 2

Fig. 11. Experiment results for minimum distance to obstacles from the nearest point fitness function.

portray diverse categories of obstacles. In our case, this problem is not relevant as we are working with static maps.

5.3.3. Experimental problems

Fig. 11 shows two solutions obtained by the algorithm when this fitness function is evaluated under the same scenario. The value of the fitness function for each solution of Fig. 11 is:

- Solution 1 = 0.00398
- Solution 2 = 0.00352

Although the value of the fitness function for the solution 2 is smaller than for solution 1, solution 2 is slightly worse as shown in Fig. 10(b). Intuitively solution 1 seems more convenient due to its smoothness, which makes it better for vehicle navigation, although there is still room for improvement in this fitness function.

Prioritizing the maximization of the minimum distance from obstacles in the occupancy map results in a lack of proportional suitability for values exceeding this minimum. Even though the trajectories are optimized according to the fitness function with the aim of staying as far away from obstacles as possible, there is a possibility that trajectories which intuitively might be considered safer for navigation due to their greater average distance from obstacles are rejected by the fitness function because they have a closer minimum distance to obstacles.

Given this situation, even if the fitness function meets the intended mathematical and conceptual requirements, it becomes imperative to explore alternatives that ensure that the trajectory maintains a maximum average distance from collisions.

5.3.4. Potential solutions

Oversimplifying the search for the trajectory with the highest minimum obstacle distance can lead to the loss of more efficient solutions. To address this, we use statistics such as the mean of distances to obtain average values that provide a more comprehensive perspective on the proximity of the trajectory to the obstacles.

5.4. AVGMD. AVeraGe minimum distance to obstacles

The AVGMD fitness function uses the average of the minimum distances of the points of the trajectory to the obstacles to find the trajectory that is as far away as possible on average from the obstacles, prioritizing avoiding collisions. The method is formally outlined in Eq. (16).

$$f_c = \frac{N}{\sum_{i=1}^{N} min_{j \in N_o} D_{ij}} \tag{16}$$

where N is the number of trajectory points, N_o is the number of obstacles in the occupancy map and D_{ij} denotes the distance from the point i to the obstacle j when there is not collision.

5.4.1. Theoretical advantages

Firstly, the approach promotes global consideration by offering an average distance perspective to obstacles while providing insights into the trajectory's proximity to obstacles. This helps prevent excessive penalties in specific areas, leading to smoother and more natural path generation.

Additionally, averaging distances along the trajectory leads to a quicker convergence of the fitness function. By reducing deviations in the trajectory, the attainment of the optimal solution is smoother. This has the potential to improve the effectiveness of optimization and planning procedures.

5.4.2. Potential drawbacks

The new fitness function requires the evaluation of some key factors. First, when dealing with dynamic or changing obstacles, relying exclusively on the mean distance may not accurately represent the current risk. This is due to the fact that the mean distance, being an average, may not sufficiently capture the evolving risk associated with moving obstacles. In our case, this is not a problem as we are working with static maps.

Similarly to previous cases, the resolution of the continuous occupancy map is a crucial factor. Opting for a low resolution can have a negative impact on the accuracy of the average distance measurement. It is essential to carefully choose a resolution that aligns with the desired level of precision and accuracy.

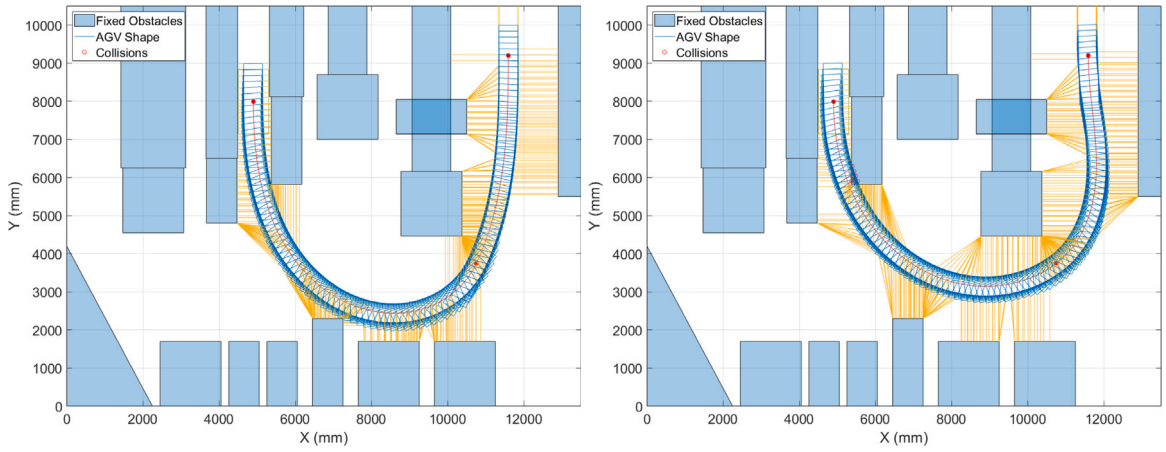
Additionally, an interesting issue arises when collisions occur, as the averaging process takes into account zero distance values (7). There may be situations where the most efficient route entails contact with objects, in opposition to preventing any collisions. Consequently, this makes difficult interpreting the average distance measurement, mainly when adversarial values emerge due to collisions.

5.4.3. Experimental problems

Fig. 12 shows two solutions obtained by the algorithm when this fitness function is evaluated under the same scenario. The value of the fitness function in each solution of Fig. 12 is:

- Solution 1 = 0.00254
- Solution 2 = 0.00166

Although the value of the fitness function of solution 2 is smaller, this solution is worse due to the fact that there are collisions as shown in



(a) Scenario 1 (b) Scenario 2

Fig. 12. Experiment results for Average minimum distance to obstacles fitness function.

Fig. 10(b). Thus, this fitness function needs to be improved to consider collisions.

When using the average distance as a parameter, it is important to consider the performance of the fitness function when there are collisions. In that case, the distance to obstacles at that point is zero, which means that collisions do not contribute to the average of minimum distances.

Using a fitness function that does not penalize collisions results in being solely dependent on the average of the minimum distances to obstacles, regardless of whether a collision scenario exists. In certain circumstances, as shown in the example, this average value may be higher for paths with collisions, even though the average distance to obstacles may be greater compared to other collision-free paths. This can cause the algorithm to converge on unsatisfactory and unacceptable solutions. In this scenario, instead of optimizing the path with a focus on safety, the fitness function may select a path with collisions as the optimal solution.

5.4.4. Potential solutions

To address the mentioned issue, we propose the use of a piece-wise function with two well-defined components: one for collision scenarios, where the invaded collision space distance is calculated, and another for collision-free scenarios, where we continue to work with the average distance of the points along the trajectory. To ensure the continuity of the piece-wise function, both components are normalized to 1.

5.5. PWAVG. Piece-Wise AVeraGe minimum distance to obstacles considering collision

The PWAVG method involves using a piece-wise function to take into account both, the average of the minimum distances from the trajectory points to the obstacles and the collision distance. This technique is expressed formally in Eq. (17).

$$f_c = \begin{cases} 1 + \sum_{i=1}^{N} min_{j \in N_o} D_{c_{ij}} & \text{if } D_c \neq 0 \\ 1 - \dfrac{\sum_{i=1}^{N} min_{j \in N_o} D_{ij}/N}{max_{j \in N_o} D_{ij}} & \text{if } D_c = 0 \end{cases} \tag{17}$$

where N is the number of trajectory points, N_o is the number of obstacles in the occupancy map, D_{ij} denotes the distance from the point i to the obstacle j when there is no collision, and $D_{c_{ij}}$ when a collision occurs. D_c is the sum of distances to all colliding obstacles. Thus, when there is no collision ($D_c = 0$), fitness function aims to maximize the distance to obstacles. On the other hand, when collisions occur ($D_c \neq 0$), the fitness function is designed to minimize the number of them.

5.5.1. Theoretical advantages

Initially, the fitness function is separated into two parts for a proper balance between collision avoidance and obstacle proximity. This approach provides a refined consideration of both direct collision and overall obstacle proximity, allowing a more effective compromise between safety and efficient navigation near obstacles.

By using a piecewise normalized function, we have stabilized the behaviour of the algorithm against changes in distances to obstacles in various scenarios. As a result, the performance of the algorithm is more robust since the cost function does not experience significant fluctuations or discontinuities. Furthermore, normalizing the fitness function to a value of 1 ensures that it provides a continuous range of values, which facilitates convergence and improves the efficiency of the optimization algorithm.

Furthermore, the fitness function provides a flexible method to assign collision penalties. Instead of using a binary penalty system, this approach calculates the total distance to the edges of objects in collision. By taking an incremental and continuous penalty approach, it allows for a more detailed treatment of collisions during optimization, resulting in a nuanced perspective on collision-related factors.

5.5.2. Potential drawbacks

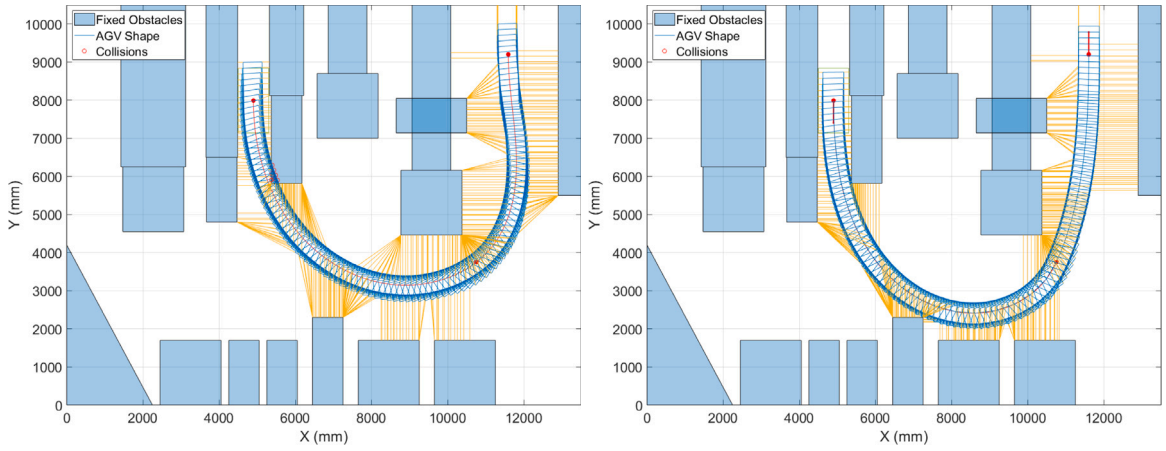
Calculating the total distance to the edges of colliding obstacles for several points along the trajectory may become computationally intensive. Moreover, it would increase proportionally to the number of obstacles in the occupancy map.

5.5.3. Experimental problems

Intensive experiments have been carried out in different scenarios and we have not identified problems. In all results a lower value in the fitness function indicates a better solution.

When using PWAVG in the experiment of Section 5.4, the results show that, as intended, using the PWAVG function the potential problems of AVGMD fitness function are avoided, being able to obtain the trajectory with the highest value of minimum average distance to obstacles, but without incurring in collisions with the occupancy map, as shown in Fig. 13.

Table 3 summarizes the key findings, providing a clear overview of the benefits and drawbacks associated with each fitness function.



(a) Scenario 1 (b) Scenario 2

Fig. 13. Experiment results comparison between AVGMD and PWAVG fitness functions.

Table 3
Summary table of the results of the proposed fitness functions.

Fitness function	Drawbacks	Advantages
MLPC	- Collision dominance and generalization - Scale sensitivity - Suboptimal optimization	- Simplicity in straightforward scenarios - Computational efficiency
MLMC	- Sensitivity of point sampling - Number of collisions uncertainty - Lack of smooth convergence gradients	- Parameter balance - Collision penalty differentiation
MDOBS	- Complex obstacle representation - Computational intensity	- Real Obstacle consideration - Penalty proportionality - Adaptability to different environments
AVGMD	- Bad mobile obstacles interaction - High resolution sensitivity	- Obstacle proximity global consideration - Smoother fitness function convergence
PWAVG	- Computational intensive proportional to map complexity	- Collision and obstacle distance balance - Collision penalties flexibility

6. Results discussion

This section presents the main evaluation criteria used to asses the fitness function of an industrial AGV trajectory optimization. For the experiments, a pulling trolley AGV is used, with dimensions of 1800 mm in length and 550 mm in width.

Based on the insights obtained from Table 3, further experimentation is carried out in different scenarios, where all proposed fitness functions are applied. This systematic approach enables us to evaluate and compare the outcomes produced by each fitness function in different trajectory optimization situations. The results are presented in a clear, colour-coded format, which illustrates the outcomes of the fitness functions.

6.1. Scenarios

Three simulation scenarios were designed to evaluate the performance of the optimization algorithms. The scenarios have the same dimensions (13,500 × 11,000 mm) and are categorized by their density levels and the arrangement of obstacles (low, medium and high). In the low complexity scenario they leave a large free zone and the design deliberately presents a dilemma for the optimization process. The algorithm must determine the optimal trajectory between the start and end points between two route options with similar clearance. The medium complexity scenario introduces a moderate increase in obstacle density. Finally, the high complexity scenario implies a positioning of obstacles that is more difficult to address and a greater number of them.

In terms of design, the occupancy map for each scenario prioritizes the strategic placement of blocks and landmarks for navigation in that environment. Fig. 14 shows the occupancy maps, where blue polygons represent obstacles and white spaces indicate free areas. The start and end points are included, along with arrows that indicate the departure angle of the trajectory at these points, which have been defined in the experiments.

6.2. Metrics

Three experiments were conducted to compare the performance of different fitness functions analysed in this study across three different complexity scenarios: low, medium, and high. This extensive evaluation provides insights into how each method performs under varying levels of complexity, allowing for a comprehensive comparison of their effectiveness in AGV trajectory optimization.

The evaluation metrics for the fitness functions include:

- Trajectory length: The distance travelled by the AGV from the starting point (x_s, y_s) to the end point (x_e, y_e)

$$s = \int_{(x_s,y_s)}^{(x_e,y_e)} \sqrt{1 + \left(\frac{dy}{dx}\right)^2}\, dx \tag{18}$$

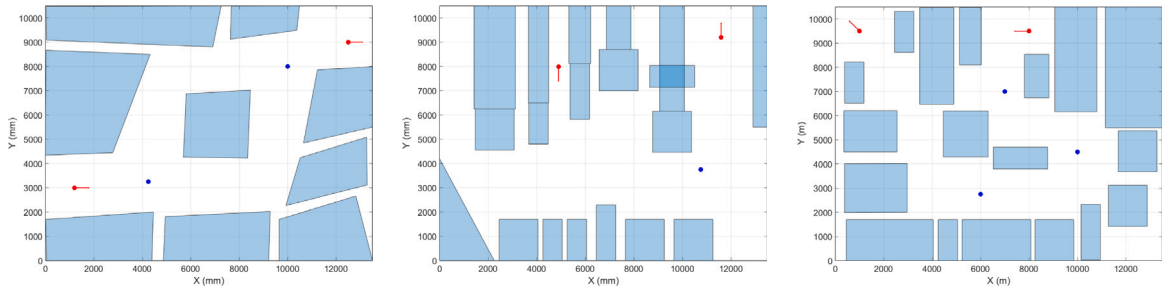


Fig. 14. Graphical representation of the three scenarios considered for optimization experiments. Red points are starting and ending point with arrows representing their output angles (**a**) Low complexity; (**b**) Medium complexity; (**c**) High complexity.

- Minimum distance to obstacles: The MDO is the minimum distance that the vertices of the AGV will have to the obstacles of the occupation map. The calculation involves determining the minimum distance to obstacles for each vertex of the vehicle along the trajectory, and selecting the smallest value from this set of values. When there is a collision registered, the MDO value is zero. Formally, this value is given by Eq. (19).

$$MDO = min_{i\in 1\ldots N,\ k\in 1\ldots 4}(min_{j\in 1\ldots N_o} D_{ijk}) \tag{19}$$

  Being N the number of trajectory points and No the number of obstacles in the occupancy map. D_{ijk} denotes the distance from the vehicle's vertex k in the trajectory point i to the obstacle j when there is no collision.
- Average distance to obstacles: The average value of the set of minimum distances to obstacles from the vertices of the vehicle along the trajectory (ADO) is formally expressed in Eq. (20). This value analyses the average distance of the trajectory to the occupation map, thus maximizing this metric means maximizing the anti-collision safety conditions.

$$ADO = \frac{\sum_{i=1}^{N} min_{j\in N_o, k\in 1..4} D_{ijk}}{N} \tag{20}$$

- Computational time: Consist of the average execution time per iteration, formalized in Eq. (21)

$$\bar{t} = \frac{1}{n_i}\sum_{i=1}^{n_i} t_i \tag{21}$$

where $\bar{t}$ is the average computational time per iteration, n_i is the total number of iterations and t_i is the computational time of an algorithm iteration.

Furthermore, a computational analysis of efficiency and demand will be conducted for each fitness function. This analysis is important for assessing the potential future application of similar methods in real-time scenarios. To measure efficiency, the average time per iteration is calculated. This is derived from the total simulation time and subsequent mean calculation. The simulations are executed in a AMD Ryzen 5 5600H with 8 GB of RAM.

The computational analysis of efficiency and demand also provides valuable insights into the practical applicability of these methods in real-time scenarios. By quantifying the computational load and efficiency of each fitness function, it can be better understood their performance characteristics and determine their suitability for deployment in dynamic operational environments.

Trajectory length, minimum obstacle distance and average obstacle distance are important indicators that provide insights into the proximity of the AGV's path to potential collisions and obstructions. By analysing these metrics, we can determine how effectively each fitness function navigates through complex environments, revealing its ability to generate trajectories that prioritize safety while optimizing efficiency.

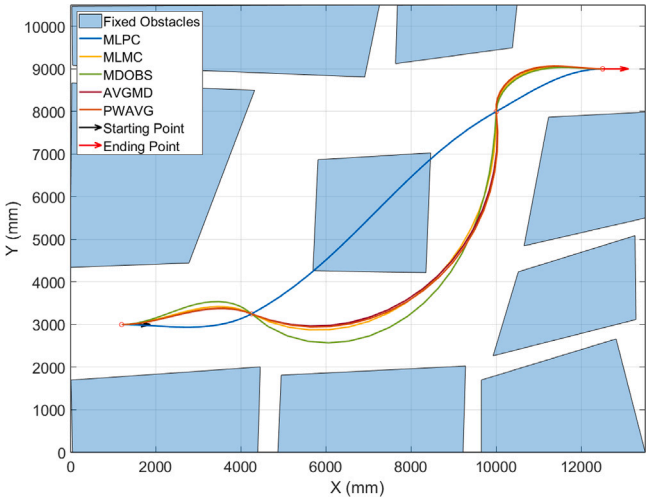


Fig. 15. Results of each fitness function in the low complexity scenario.

6.3. Low complexity results

In this type of scenarios, it is about characterizing the trajectories, detecting collision points and seeing how the fitness function behaves with a map with a low density of obstacles.

Fig. 15 shows the trajectories generated with the different fitness functions since there are several possible routes. As seen in the figure, the MLPC function is not capable of detecting collisions that occur during trajectory tracking, and leads the algorithm to obtain the shortest trajectory without trying to avoid obstacles since it is based on the error. The MDOBS function is most differentiated from the others in that it aims to maximize the minimum distance to obstacles, so the trajectory deviates as much as possible from the corner of the central obstacle on the map, instead of gravitating towards the obstacles in previous areas. The remaining cost functions, although slightly different, produce similar collision-free trajectories.

Fig. 16 presents a comparative analysis detailing the minimum distances to obstacles on the occupancy map along the trajectory. Collisions are represented with a red circle (distance less than zero), which provides an overview of how the fitness functions work differently. Both the MLPC and AVGMD functions record collisions in their path; MDOBS shows the greatest deviation from the middle obstacle, while the rest of the functions (MLMC, AVGMD and PWAVG) show very similar minimum distances, with AVGMD being the one that collides with the middle obstacle.

As it is difficult to visually determine the best fitness function, analytical metrics have been evaluated and are shown in Table 4.

Table 4
Analytic results of the different fitness functions in the low complexity scenario.

Fit. function	Length (mm)	Avg. distance (mm)	Min. distance (mm)	Collisions	Comp. time (s)
MLPC	$1.323\,94 \times 10^4$	31.17	0	28	2.362
MLMC	1.5167×10^4	535.35	4.38	0	2.550
MDOBS	1.5599×10^4	494.15	169.57	0	2.518
AVGMD	1.5152×10^4	541.97	0	1	2.574
PWAVG	1.5223×10^4	542.25	17.86	0	2.587

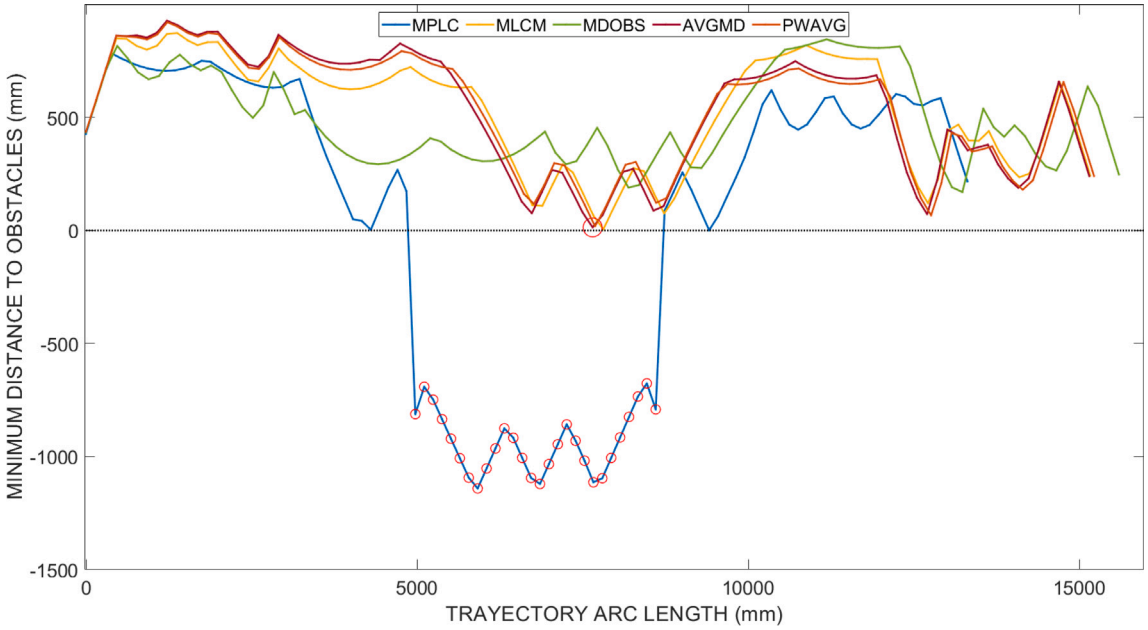


Fig. 16. Comparison of minimum distance to obstacles for the different fitness functions evaluated in the low complexity scenario.

These results show that PWAVG maintains the greatest average distance to obstacles, despite not having the greatest minimum distance to them. However, MDOBS surpasses it by minimizing the minimum distance. On the other hand, MDOBS presents the highest minimum distance although with a lower average distance to obstacles. As shown in Table 4, the execution times for each iteration with the different cost functions are practically identical. It should be noted that as the complexity of the aptitude function increases there is a slight increase in execution time.

6.4. Medium complexity results

In the medium complexity scenario, the trajectories calculated by the five fitness functions studied are shown in Fig. 17. It can be seen that MLPC generates a trajectory with many collisions; AVGMD, although generating a very efficient trajectory in terms of minimum average distance, also produces a large number of collisions in the most complicated part of the trajectory. On the contrary, MLMC, MDOBS and PWAVG produce trajectories comparable to each other, with PWAVG being the only one capable of finding a route without collisions with the environment.

Collisions with objects can be examined by analysing the minimum distances between the vehicle and obstacles shown in Fig. 18. The x-axis indicates the distance travelled from the starting point of the trajectory. Collisions are shown with a red circle (distance less than zero)

From Fig. 18 it can be deduced that the only collision-free trajectory is the one calculated by PWAVG, and the total number of collisions of the other trajectories can be determined. It is possible to visually estimate which trajectories have the longest average distance and their minimum distance to obstacles. Sharp spikes and fluctuations are

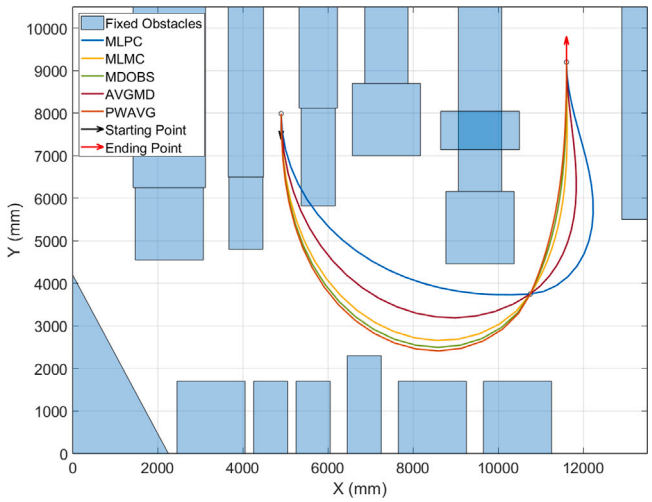


Fig. 17. Results of each fitness function in the medium complexity scenario.

caused by the trajectory getting closer to one obstacle than another on the occupancy map. As a result, the tendency to move away from or towards the obstacle changes. These sudden changes occur more frequently on occupancy maps with a greater number of obstacles

As a summary, Table 5 shows the metrics of each fitness function. MLPC has the shortest path but also the highest number of collisions. On the other hand, AVGMD has the highest average minimum obstacle distance, but it is not feasible in scenarios like the one studied due to the detection of 12 collisions. The PWAVG fitness function is the only

Table 5
Analytic results of the different fitness functions in the medium complexity scenario.

Fit. function	Length (mm)	Avg. distance (mm)	Min. distance (mm)	Collisions	Comp. time (s)
MLPC	1.4374×10^4	332.67	0	19	4.139
MLMC	1.5334×10^4	481.69	0	1	4.145
MDOBS	1.5706×10^4	422.20	0	1	4.156
AVGMD	1.4691×10^4	584.86	0	12	4.180
PWAVG	1.5708×10^4	391.04	2.82	0	4.193

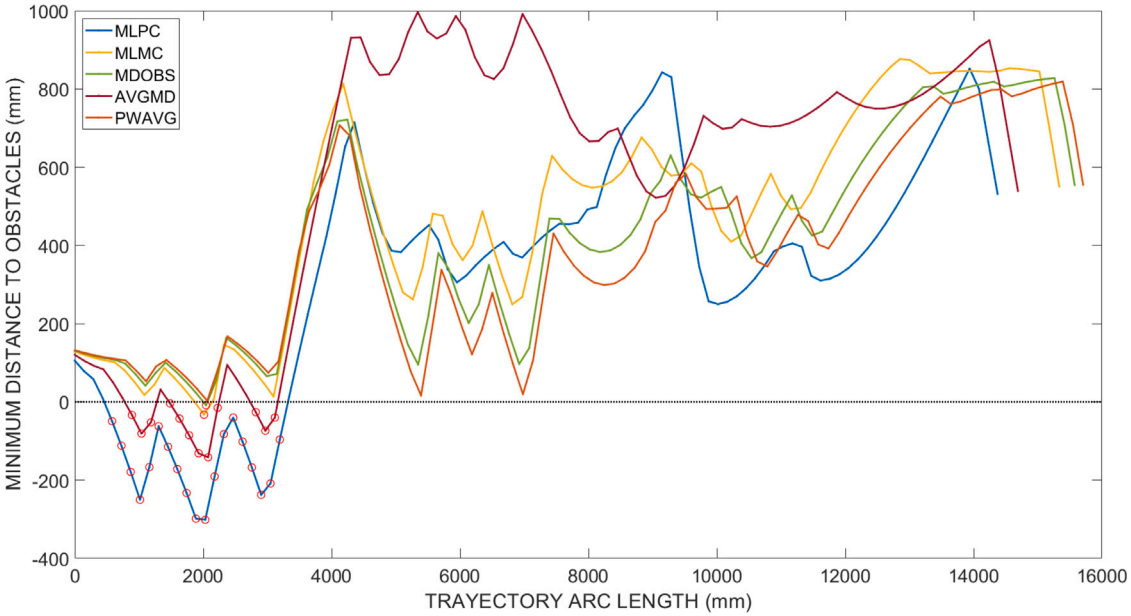


Fig. 18. Comparison of minimum distance to obstacles for the different fitness functions evaluated in the medium complexity scenario.

one that is collision-free, confirming its good performance. To determine collisions, a total of 100 trajectory samples are taken. Table 5 also indicates a great similarity between the execution times of iterations with the different fitness functions. However, it is worth noting that with increasing complexity of the cost function there is a slight increase in execution time.

6.5. High complexity results

The high complexity scenario stands out for the greater density of obstacles and greater difficulty in finding a route without collisions. Fig. 19 shows the trajectories calculated for each cost function. AVGMD, with the objective of maximizing the average distance, suffers a collision towards the end of the trajectory, despite maintaining a well-centred trajectory between obstacles. In Fig. 20, an analysis of the collision behaviour and distance to the obstacle of each trajectory is carried out. It is observed that all the trajectories get dangerously close to the obstacles towards the end, with MPLC, MDOBS and AVGMD being the ones that collide with the obstacles on the occupancy map, as indicated by the red dots. The MLPC trajectory also shows collisions in the midsection of the trajectory as it seeks the shortest path despite the risk of collision. It is worth noting that AVGMD and PWAVG have the highest average distance despite suffering from collisions. The reason for the collision risk of the MLPC is the minimum distance to obstacles at the beginning of the trajectory, which contrasts with its acceptable minimum distance in the middle section of the route.

Table 5 summarizes the metrics of each fitness function. It is observed that only MLMC and PWAVG are capable of generating collision-free trajectories. PWAVG maintains a greater average and minimum distance to obstacles. Furthermore, PWAVG produces the shortest path length, second only to MLPC, which prioritizes achieving the shortest

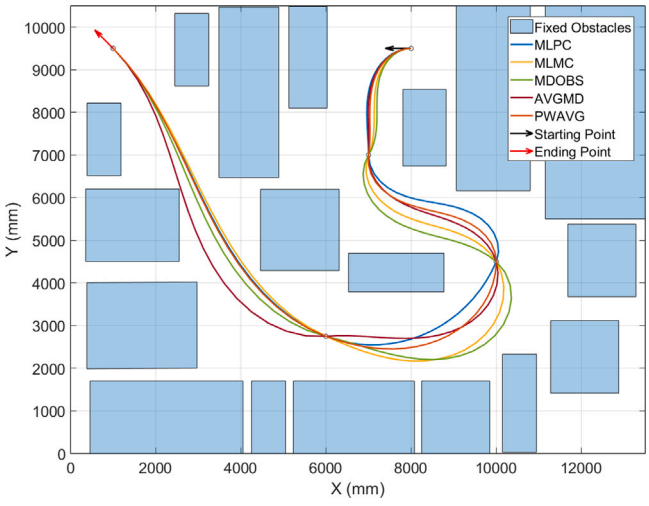


Fig. 19. Results of each fitness function in the high complexity scenario.

collision-free distance but incurs collisions during its path. Table 6 shows the iteration execution times for the fitness functions with the trend seen previously. There is a slight increase in execution time as the complexity of the cost function increases.

The results show that PWAVG generates collision-free trajectories in all three scenarios, outperforming other alternatives in terms of both analytical and visual outcomes. However, trajectories that yield better metrics but exhibit collisions along their paths are not desirable. These outcomes were expected due to the design process aimed at improving

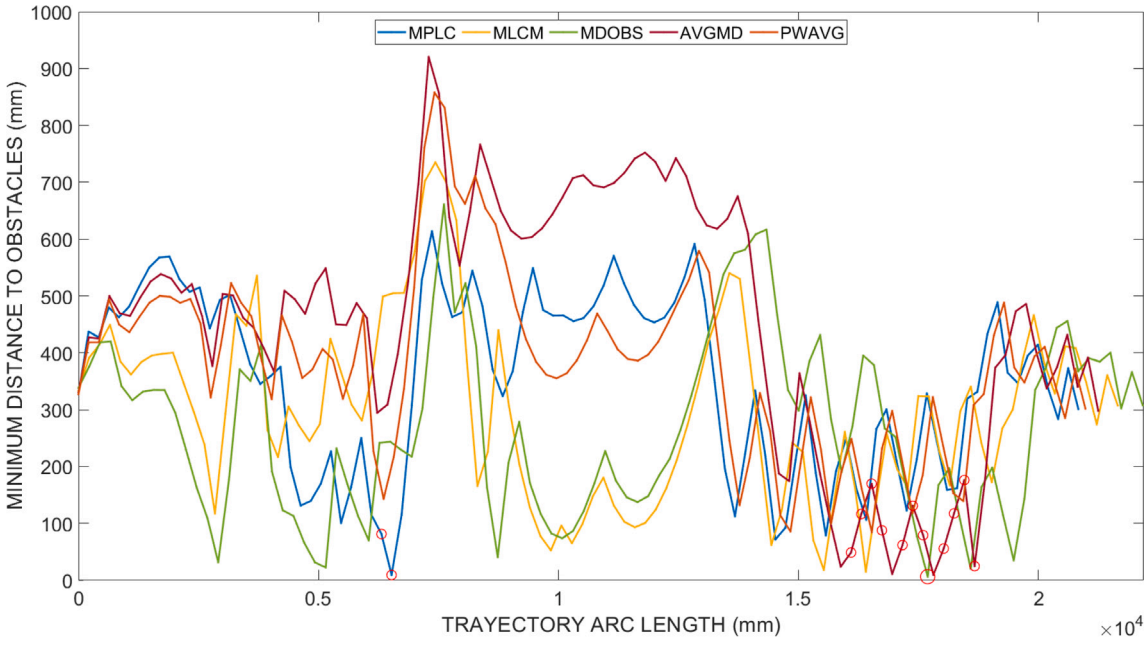


Fig. 20. Comparison of minimum distance to obstacles for the different fitness functions evaluated in the high complexity scenario.

Table 6
Analytic results of each evaluation metric for the high complexity scenario.

Fit. function	Length (mm)	Avg. distance (mm)	Min. distance (mm)	Collisions	Comp. time (s)
MLPC	2.0835×10^4	353.12	0	2	8.312
MLMC	2.1657×10^4	299.25	14.93	0	8.699
MDOBS	2.2176×10^4	268.60	0	1	8.750
AVGMD	2.1213×10^4	449.94	0	11	8.970
PWAVG	2.0983×10^4	384.85	8.282	0	8.977

the fitness function. In conclusion, PWAVG is a standout performer in terms of safety and efficiency in trajectory generation, highlighting its adaptability across various simulation scenarios. Table 6 shows the iteration execution times for the analysed fitness functions too, indicating a consistent trend. However, there is a slight increase in execution time as the complexity of the fitness function increases. This correlation suggests that the complexity of the fitness function may affect the efficiency of the optimization process, especially in more demanding scenarios.

6.6. Comparison with existing methods

Knowledge of the strengths and weaknesses of the different approaches for AGV trajectory optimization is important not only to validate the proposals but also for future research and improvement efforts. In this section we compare the proposed fitness function with that used by Lamini et al. (2018) for AGV trajectory optimization.

In their publication, Lamini et al. (2018) suggest an improved crossover operator to address path planning using genetic algorithms (GA). It is applied in a static environment segmented into cells of uniform size where the occupancy status is indicated. The fitness function evaluates the trajectory quality based on several criteria, such as path length, safety levels, and energy consumption, and it is expressed in Eq. (22).

$$f_L(p) = \frac{1}{w_l l(p) + w_{s_1} s_1(p) + w_{s_2} s_2(p)} - e \tag{22}$$

where $l(p)$ is the trajectory length. $s_1(p)$ is the Safety First Level (SFL) calculated as $s_1(p) = \sum_{i=1}^{N-1} s_i$. In this equation s_i receives a penalty of (+1) if an obstacle is in the first security level of distance to the current position (x_i, y_i) of the vertices of the AGV. $s_2(p)$ is the Safety Second Level (SSL). This value is calculated as $s_2(p) = \sum_{i=1}^{N-1} s_i.s_i$ receives a penalty of (+1) if an obstacle is in the second security level of distance to the current position (x_i, y_i) of the vertices of the AGV. e is the energy penalty that receives a penalty of (+1) if the robot turns, and (0) if it continues forward and w_l, w_{s1}, and w_{s2} are the weights of the length, SFL, and SSL properties for a given path p.

However, it is important to note that our parameter selection approach differs from that of Lamini et al. They consider energy consumption and path length. In our case, taking into account the specific requirements for optimizing the AGV trajectory, energy consumption has not been taken into account and other factors such as distance to obstacles and efficiency have been prioritized. In addition, the inverse of the function has been used to perform the optimization process by minimizing its value. The resulting fitness function used in this comparison is expressed in Eq. (23).

$$f_L(p) = w_{s_1} s_1(p) + w_{s_2} s_2(p) \tag{23}$$

For each scenario, simulations of 3600 s are carried out with both functions. Finally, limitation of both methods have been addressed and compared.

6.6.1. Low complexity scenario

Fig. 21 shows the results obtained in the low complexity scenario, with PWAVG (left) and Lamini's (right). As can be seen, there are some differences in trajectory planning with the two methods. With PWAVG it appears that the vehicle moves closer to the obstacle since this algorithm prioritizes maintaining a high average distance. However, the fitness function proposed by Lamini results in a trajectory that maintains a greater distance to the midpoint of the obstacle, although it is closer to the occupancy map in the previous area. Both solutions are valid, PWAVG maintains a higher average distance, but the minimum

Table 7
Analytic results of PWAVG and Lamini's optimization in the low complexity scenario.

Fitness function	Length (mm)	Avg. distance (mm)	Min. distance (mm)	Collisions	Time/Iteration (s)
PWAVG	1.5223×10^4	542.25	17.86	0	2.587
Lamini et al. (2018)	1.5754×10^4	497.76	101.21	0	2.244

Table 8
Analytic results of PWAVG and Lamini's optimization in the medium complexity scenario.

Fitness function	Length (mm)	Avg. distance (mm)	Min. distance (mm)	Collisions	Time/Iteration (s)
PWAVG	1.5708×10^4	391.04	2.82	0	4.193
Lamini et al. (2018)	1.5188×10^4	520.85	0	2	4.041

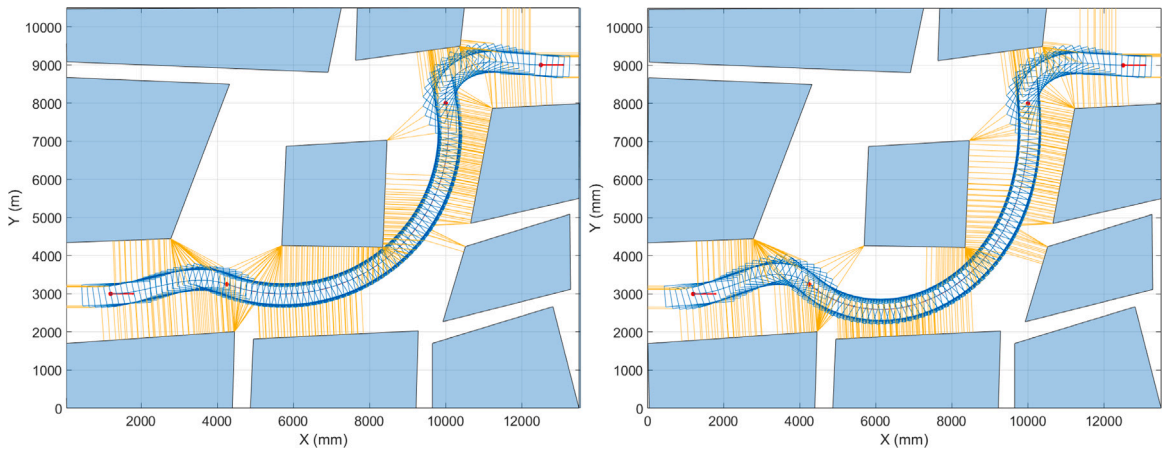


Fig. 21. Comparison of AGV trajectory optimization using (a) PWAVG (left) and (b) Lamini's (Lamini et al., 2018) fitness functions, for low complexity scenario.

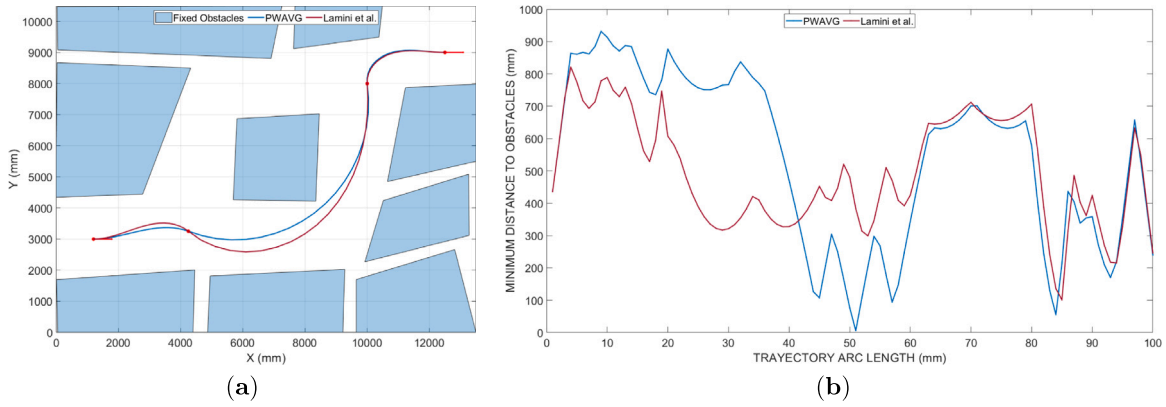


Fig. 22. Comparison of AGV trajectory optimization using PWAVG and Lamini's (Lamini et al., 2018) fitness functions, for low complexity scenario. (a) Trajectories. (b) Minimum distance to obstacles.

distance can be smaller than with that of Lamini et al. The detailed results of the evaluated metrics are shown in Table 7.

Fig. 22a shows the graphic comparison of the two trajectories obtained with each of the cost functions. The results discussed above are confirmed regarding the average distance and distance to obstacles. The spikes and sudden changes that appear in Fig. 22b indicate that the trajectory is approaching one obstacle more than another on the occupancy map, causing a directional change away from the obstacle. These changes occur more frequently in occupancy maps with higher density of obstacles.

6.6.2. Medium complexity scenario

Fig. 23 shows the results of the simulations carried out with the PWAVG and Lamini's fitness functions in the medium complexity scenario, where there is a moderate increase in the density of obstacles compared to the previous one. In this medium complexity scenario, the PWAVG fitness function causes the vehicle to pass very close to one of the obstacles. This is because the starting point is located in a densely obstructed area, which poses a great challenge in navigating the waypoints without collisions. But the PWAVG function is capable of finding a collision-free path, although approaching the bottom of the map. In contrast, the fitness function of Lamini et al. it fails to avoid collisions because it lacks collision detection mechanisms and relies solely on penalties for proximity to obstacles. As a result, the trajectory tries to stay as far away from the bottom region of the map as possible, causing collisions in the initial stage. In this case PWAVG is superior when it comes to prioritizing safety and effectively avoiding collisions. The detailed results of the metrics are shown in Table 8.

Fig. 24a presents the two trajectories obtained. Fig. 24b shows that although the trajectories may appear very similar, the fitness function of Lamini et al. produces a collision while PWAVG avoids obstacles, making it a safer option in this scenario.

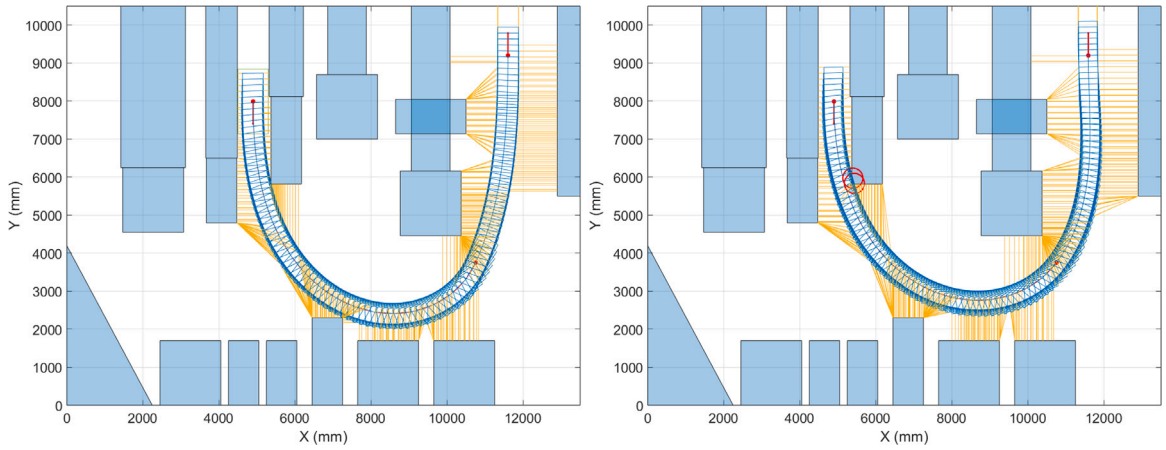


Fig. 23. Comparison of AGV trajectory optimization using (a) PWAVG (left) and (b) Lamini's (Lamini et al., 2018) fitness functions, for medium complexity scenario.

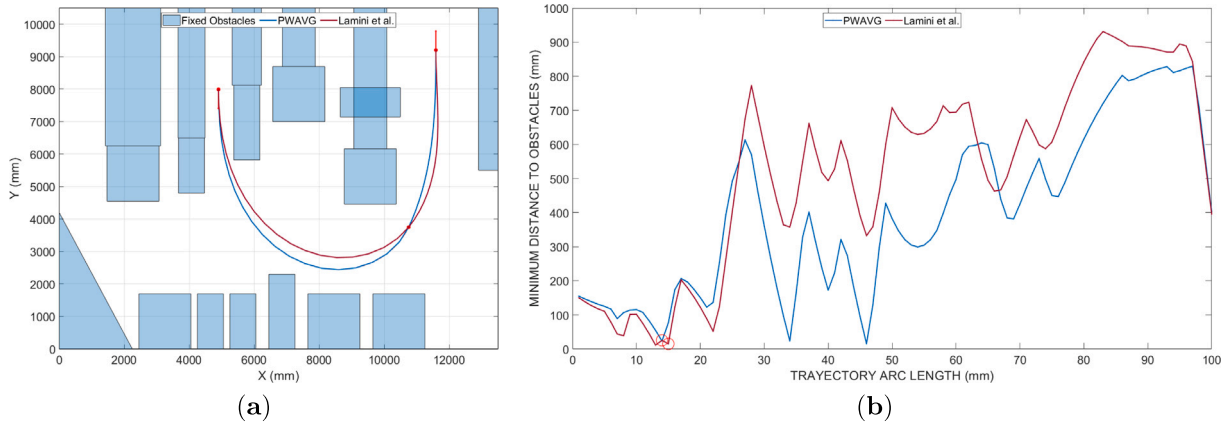


Fig. 24. Comparison of AGV trajectory optimization using PWAVG and Lamini's (Lamini et al., 2018) fitness functions, for medium complexity scenario. (a) Trajectories. (b) Minimum distance to obstacles.

Table 9
Analytic results of PWAVG and Lamini's optimization in the high complexity scenario.

Fitness function	Length (mm)	Avg. distance (mm)	Min. distance (mm)	Collisions	Time/Iteration (s)
PWAVG	2.0983×10^4	384.85	8.282	0	8.977
Lamini et al. (2018)	2.0833×10^4	350.51	0	1	8.632

6.6.3. High complexity scenario

The high complexity scenario offers a map with strategically located obstacles and difficult-to-follow reference points, in order to evaluate the optimization results with the two proposals, as shown in Fig. 25.

In this case, a situation similar to that of the medium complexity scenario occurs. The fitness function of Lamini et al. produces a very satisfactory result, but causes a collision with an obstacle towards the end of the trajectory. Table 9 presents the metrics that demonstrate that the solution proposed by this fitness function has a shorter average distance than the PWAVG solution, even though the solution of Lamini et al. appears to be further from the obstacles on the occupancy map at first glance. These results, along with the occurrence of collisions, highlight the superior performance of PWAVG even in high complexity scenarios.

Fig. 26a shows the trajectories obtained, validating the analytical examination. Fig. 26b illustrates that only the PWAVG trajectory remains collision-free, in contrast to the collision-prone trajectories generated by the other function. Although the trajectories show notable similarity, the key difference is that the fitness function of Lamini et al. produces a collision that PWAVG is able to avoid, making it the safest option in this scenario. Upon closer inspection of the figures, one can discern trajectories with the longest average distance to obstacles and those in closest proximity to them.

6.6.4. Limitations of compared fitness function in trajectory planning

By comparing the performance of PWAVG and the fitness function proposed by Lamini et al. In different complexity scenarios the following can be concluded. In the low complexity scenario, PWAVG maintains a larger average distance to obstacles, while the function of Lamini et al. shows a longer minimum distance but with a trajectory approach to obstacles in certain sections. In the medium complexity scenario, PWAVG manages to avoid collisions. In the high complexity scenario, PWAVG not only shows a larger average distance but also performs better than Lamini et al. since it effectively avoids collisions. These findings highlight the efficiency and usefulness of PWAVG in AGV route optimization scenarios of different levels of complexity.

Continuing with the comparison of the two fitness functions, PWAVG and Lamini's, the limitations of the latter are exposed. Lamini's fitness feature does not directly detect collisions, but rather penalizes proximity to obstacles at two different levels. Consequently, in some situations the fitness function may require the trajectory to maintain a constant distance to obstacles at all points, which may result in the vehicle being inevitably close to obstacles and incurring penalties.

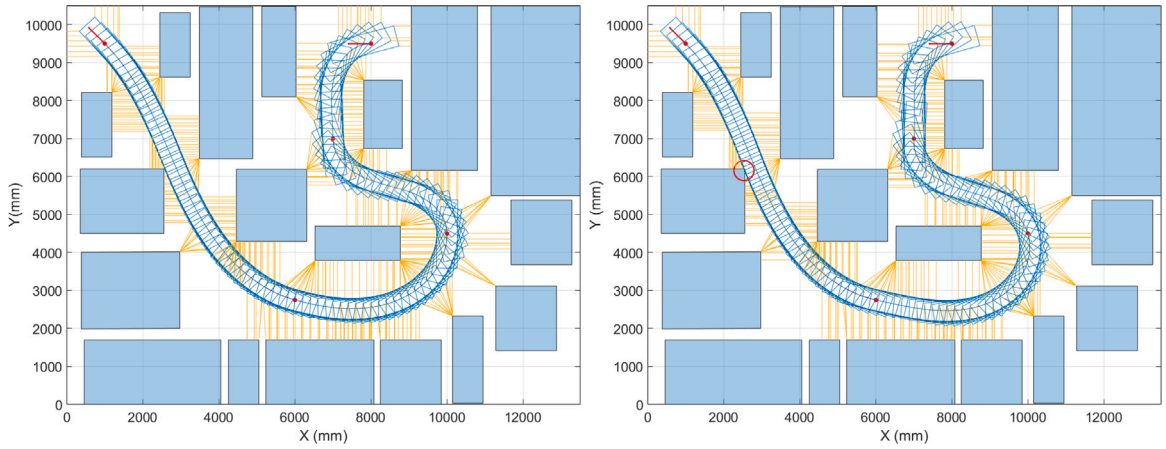


Fig. 25. Comparison of AGV trajectory optimization using (a) PWAVG (left) and (b) Lamini's (Lamini et al., 2018) fitness functions, for high complexity scenario.

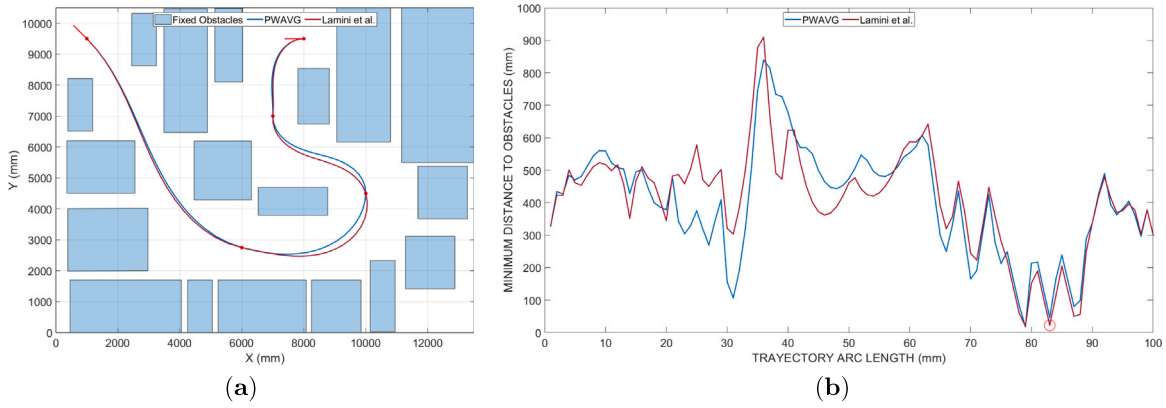


Fig. 26. Comparison of AGV trajectory optimization using PWAVG and Lamini's (Lamini et al., 2018) fitness functions, for high complexity scenario. (a) Trajectories. (b) Minimum distance to obstacles.

Table 10
Analytic results of Lamini et al. (2018) fitness functions for two different scenarios illustrating its limitations in detecting collisions and optimizing trajectories.

Scenario	Length (mm)	Avg. distance (mm)	Min. distance (mm)	Collisions	Fitness function value
a	1.5708×10^4	520.85	0	2	15.55
b	1.5616×10^4	409.72	19.55	0	17.15

Therefore, the fitness function directs the trajectory to distance itself as much as possible from other obstacles, which can lead to collisions in areas with obstacles in close proximity to each other. This behaviour is unacceptable since the fitness function can consider an optimal trajectory in which the vehicle collides with its surroundings.

Fig. 27 presents two trajectories: scenario A, which represents the trajectory generated by Lamini's fitness function, and scenario B, which represents a collision-free trajectory under identical conditions. The analytical data in Table 10 shows that Lamini's produces a higher value in the no-collision scenario. During the optimization process, the algorithm considers the collision-free path to be less safe based on the fitness function and discards it. This contradicts the security-oriented approach adopted in this study and highlights the superior performance of the PWAVG function in that regard.

7. Conclusions and future works

In this work, the definition of an effective fitness function, the key component of an optimization method, is discussed in detail. The fitness function of a GA quantifies the quality of an industrial automated vehicle trajectory for different scenarios with obstacles.

Throughout the fitness function design process, we have identified and addressed several challenges that impact accuracy and suitability for specific problem requirements. These challenges include considerations such as path length, obstacle spacing and collision minimization within the occupancy map.

In particular, a key challenge is the sensitivity of the fitness function parameters. Algorithm efficiency is significantly influenced by parameter scaling and tuning, showing the importance of precise tuning for optimal results.

Multi-factor optimization has also emerged as a difficult task. A delicate balance between fitness function components is often required to ensure an accurate representation of the problem goals and constraints.

Another fundamental issue is the premature convergence of the genetic algorithm. If the fitness function is not adequately defined, there is a risk that the algorithm will settle on sub-optimal solutions before thoroughly exploring the search space.

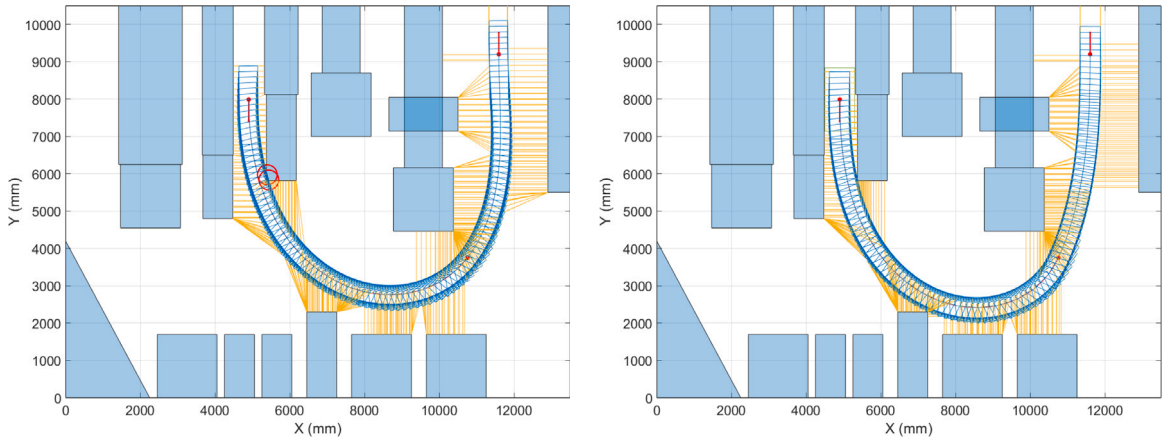


Fig. 27. Example scenario illustrating the limitations of Lamini et al. (2018) fitness function in detecting collisions. (a) Case with collisions; (b) Case without collisions.

To effectively address these challenges, different approaches have been proposed. These include rigorously adjusting the weights of the fitness function terms to increase the flexibility of the optimization. Furthermore, it is emphasized the need to properly tune the optimization parameters according to the specific problem requirements, which significantly influences the convergence towards optimal solutions.

Finally, we can highlight the importance of intelligent fitness function design, using prior knowledge to ensure an accurate representation of the problem objectives and constraints. This reduces the likelihood of premature convergence and contributes to the overall effectiveness of genetic algorithms in tackling complex optimization problems.

In summary, this article deals with the importance of the fitness function in genetic algorithm optimization and provides a detailed insight into the associated challenges of its definition and tuning. It draws valuable recommendations for overcoming these challenges, making a significant contribution to the field of optimization in complex problems for industrial applications.

The method presented in this work has been proven effective in real scenarios in a static environment. In fact, it has always been able to find a feasible solution. The main limitation of this method in general lies in the technical limitations, particularly in the adaptation of the method to real-time scenarios with dynamic maps, where high computational power is required to perform complex optimization processes within a sufficiently short period of time to adapt to changes in the environment. Possible solutions to this limitation could be considered slow dynamic changes in the environment, using more powerful hardware, or performing calculations remotely on high-performance computing equipment.

Incorporating probability functions into the fitness function shows potential to improve trajectory planning algorithms. They allow modelling the uncertainty of the environment, the dynamics of the system and user preferences. This is a promising research direction to promote continuous progress in AGV path planning approaches. Future research will explore the integration of a Friedman test and other statistical analysis approaches to develop upcoming fitness functions, particularly with a larger sample size (Zamri et al., 2022) and Manoharam et al. (2023).

It could also be interesting to adapt the offline method proposed here to dynamic environments. This involves ensuring stability and predictability in AGV operations in the face of changing conditions. Finally, given the prevalence of real-world problems involving the optimization of conflicting objectives, it is of great interest to explore multi-objective optimization approaches and design effective fitness functions to balance these objectives.

CRediT authorship contribution statement

Eduardo Bayona: Writing – original draft, Software, Investigation, Formal analysis, Data curation, Conceptualization. **J. Enrique Sierra-García:** Investigation, Formal analysis, Conceptualization, Methodology, Supervision, Writing – original draft. **Matilde Santos:** Conceptualization, Investigation, Supervision, Writing – review & editing. **Ioannis Mariolis:** Conceptualization, Investigation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work was partially supported by the European Commission under the European Projects CoLLaboratE (grant number 820767) and MANiBOT (grant number 101120823).

References

Abdessemed, F., Faisal, M., Emmadeddine, M., Hedjar, R., Al-Mutib, K., Alsulaiman, M., Mathkour, H., 2014. A hierarchical fuzzy control design for indoor mobile robot. Int. J. Adv. Robot. Syst. 11 (3), 33. http://dx.doi.org/10.5772/57434.

Al-Sagban, M., Dhaouadi, R., 2012. Neural-based navigation of a differential-drive mobile robot. In: 2012 12th International Conference on Control Automation Robotics & Vision. ICARCV, pp. 353–358. http://dx.doi.org/10.1109/ICARCV.2012.6485184.

Alencar, H., Santos, W., Neto, G., 2022. Differential Geometry of Plane Curves, No. V. 96 in Student Mathematical Library. American Mathematical Society, Providence, Rhode Island.

Alouache, A., Wu, Q., 2018. Genetic algorithms for trajectory tracking of mobile robot based on pid controller. In: IEEE 14th International Conference on Intelligent Computer Communication and Processing. ICCP, pp. 237–241. http://dx.doi.org/10.1109/ICCP.2018.8516587.

Amiridis, K., Psarianos, B., 2015. 3-d road design by applying differential geometry. Math. Des. Tech. Aesthetics 3, 46–75.

Ardiyanto, I., Miura, J., 2012. Real-time navigation using randomized kinodynamic planning with arrival time field. Robot. Auton. Syst. 60 (12), 1579–1591. http://dx.doi.org/10.1016/j.robot.2012.09.011.

Bayona, E., Sierra-García, J.E., Santos, M., 2023a. Generation of optimum frenet curves by genetic algorithms for agvs. In: Maglogiannis, I., Iliadis, L., MacIntyre, J., Dominguez, M. (Eds.), In: Artificial Intelligence Applications and Innovations, vol. 676, Springer Nature Switzerland, Cham, pp. 454–464. http://dx.doi.org/10.1007/978-3-031-34107-6_36.

Bayona, E., Sierra-García, J.E., Santos, M., 2023b. Optimization of trajectory generation for automatic guided vehicles by genetic algorithms. In: García Bringas, P., Pérez García, H., Martinez-de Pison, F.J., Villar Flecha, J.R., Troncoso Lora, A., de la Cal, E.A., Herrero, Á., Martínez Álvarez, F., Psaila, G., Quintián, H., Corchado Rodriguez, E.S. (Eds.), 17th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2022). Springer Nature Switzerland, Cham, pp. 484–492.

Bertolazzi, E., Frego, M., 2015. G1 fitting with clothoids. Math. Methods Appl. Sci. 38 (5), 881–897. http://dx.doi.org/10.1002/mma.3114.

Candido, S., Kim, Y.-T., Hutchinson, S., 2008. An improved hierarchical motion planner for humanoid robots. In: Humanoids 2008-8th IEEE-RAS International Conference on Humanoid Robots. pp. 654–661. http://dx.doi.org/10.1109/ICHR.2008.4756021.

Chen, X., Kong, Y., Fang, X., Wu, Q., 2011. A fast two-stage ACO algorithm for robotic path planning. Neural Comput. Appl. 22 (2), 313–319. http://dx.doi.org/10.1007/s00521-011-0682-7.

Chen, Z., Xiong, G., Liu, S., Shen, Z., Li, Y., 2022. Path planning of mobile robot based on an improved genetic algorithm. In: IEEE 2nd International Conference on Digital Twins and Parallel Intelligence. DTPI, pp. 1–6. http://dx.doi.org/10.1109/DTPI55838.2022.9998894.

Corominas Murtra, A., Trulls, E., Sandoval, O., Pérez-Ibarz, J., Vasquez, D., Mirats-Tur, J.M., Ferrer, M., Sanfeliu, A., 2010. Autonomous navigation for urban service mobile robots. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4141–4146. http://dx.doi.org/10.1109/IROS.2010.5649151.

Cosío, F.A., Castañeda, M.P., 2004. Autonomous robot navigation using adaptive potential fields. Math. Computer Model. 40 (9–10), 1141–1156.

Dai, R., Cochran, J.E., 2009. Path planning for multiple unmanned aerial vehicles by parameterized cornu-spirals. In: 2009 American Control Conference. pp. 2391–2396. http://dx.doi.org/10.1109/ACC.2009.5159914.

Englot, B., Hover, F., 2011. Multi-goal feasible path planning using ant colony optimization. In: 2011 IEEE International Conference on Robotics and Automation. pp. 2255–2260. http://dx.doi.org/10.1109/ICRA.2011.5980555.

Han, Y., Zhang, L., Tan, H., Xue, X., 2019. Mobile robot path planning based on improved particle swarm optimization. In: Chinese Control Conference. CCC, pp. 4354–4358. http://dx.doi.org/10.23919/ChiCC.2019.8866634.

Hu, Y., Yang, S., 2004. A knowledge based genetic algorithm for path planning of a mobile robot. In: IEEE International Conference on Robotics and Automation, 2004. Proceedings, Vol. 5. ICRA '04, pp. 4350–4355. http://dx.doi.org/10.1109/ROBOT.2004.1302402.

Ilin, V., Simić, D., Simić, S.D., Simić, S., Saulić, N., Calvo-Rolle, J.L., 2022. A hybrid genetic algorithm, list-based simulated annealing algorithm, and different heuristic algorithms for the travelling salesman problem. Log. J. IGPL 31 (4), 602–617. http://dx.doi.org/10.1093/jigpal/jzac028.

Iser, R., Wahl, F.M., 2010. Antslam: Global map optimization using swarm intelligence. In: 2010 IEEE International Conference on Robotics and Automation. pp. 265–272. http://dx.doi.org/10.1109/ROBOT.2010.5509254.

Kang, Y.H., Lee, M.C., Kim, C.Y., Yoon, S.M., Noh, C.B., 2011. A study of cluster robots line formatted navigation using potential field method. In: 2011 IEEE International Conference on Mechatronics and Automation. pp. 1723–1728. http://dx.doi.org/10.1109/ICMA.2011.5986370.

Kreyszig, E., 2011. Advanced Engineering Mathematics, tenth Edition Wiley, Hoboken, NJ.

Kusuma, Z.D.S., Indriawati, K., Widjiantoro, B.L., Hija, A.I., Nurhadi, H., 2022. Optimal trajectory planning generation for autonomous vehicle using frenet reference path. In: 2022 5th International Seminar on Research of Information Technology and Intelligent Systems. ISRITI, pp. 480–484. http://dx.doi.org/10.1109/ISRITI56927.2022.10052833.

Ladd, A., Kavraki, L., 2004. Measure theoretic analysis of probabilistic path planning. IEEE Trans. Robot. Autom. 20 (2), 229–242. http://dx.doi.org/10.1109/TRA.2004.824649.

Lamini, C., Benhlima, S., Elbekri, A., 2018. Genetic algorithm based approach for autonomous mobile robot path planning. Procedia Comput. Sci. 127, 180–189. http://dx.doi.org/10.1016/j.procs.2018.01.113.

Lee, J., Kwon, O., Zhang, L., Yoon, S.-E., 2014. A selective retraction-based rrt planner for various environments. IEEE Trans. Robot. 30 (4), 1002–1011. http://dx.doi.org/10.1109/TRO.2014.2309836.

Liu, H., Huang, S., Gai, Y., 2010a. Robot continuous trajectory planning based on frenet-serret formulas. In: International Conference on Computer, Mechatronics, Control and Electronic Engineering, Vol. 4. pp. 47–51. http://dx.doi.org/10.1109/CMCE.2010.5610238.

Liu, H., Wan, W., Zha, H., 2010b. A dynamic subgoal path planner for unpredictable environments. In: IEEE International Conference on Robotics and Automation. pp. 994–1001. http://dx.doi.org/10.1109/ROBOT.2010.5509324.

Liu, H., Zhang, X., Gao, H., Yuan, J., Chen, X., 2021. Robust localization and map updating based on euclidean signed distance field map in dynamic environments. In: ACM International Conference Proceeding Series. pp. 85–90. http://dx.doi.org/10.1145/3462648.3462664.

Mac, T.T., Copot, C., Tran, D.T., De Keyser, R., 2016. Heuristic approaches in robot path planning: A survey. Robot. Auton. Syst. 86, 13–28. http://dx.doi.org/10.1016/j.robot.2016.08.001.

Manoharam, Gaeithry, Kasihmuddin, Mohd Shareduwan Mohd, Antony, Siti Noor Farwina Mohamad Anwar, Romli, Nurul Atiqah, Rusdi, Nur 'Afifah, Abdeen, Suad, Mansor, Mohd Asyraf, 2023. Log-linear-based logic mining with multi-discrete hopfield neural network. Mathematics 11 (9), 2121. http://dx.doi.org/10.3390/math11092121.

Martins, G.D.M., Naruto, I.d.L., Danner, P., Frencl, V.B., 2018. A trajectory simulator using frenet–serret formulas applied to punctual objects. In: 13th IEEE International Conference on Industry Applications. INDUSCON, pp. 750–755. http://dx.doi.org/10.1109/INDUSCON.2018.8627338.

Masehian, E., Sedighizadeh, D., 2007. Classic and heuristic approaches in robot motion planning - a chronological review, world academy of science, engineering and technology 29.

Mester, G., 2008. Obstacle avoidance and velocity control of mobile robots. In: 2008 6th International Symposium on Intelligent Systems and Informatics. pp. 1–5. http://dx.doi.org/10.1109/SISY.2008.4664918.

Milos, S., 2007. Roadmap methods vs. cell decomposition in robot motion planning. In: 6th WSEAS International Conference on Signal Processing, Robotics and Automation. pp. 127–132.

Patle, B., Pandey, A., Parhi, D., Jagadeesh, A., et al., 2019. A review: On path planning strategies for navigation of mobile robot. Def. Technol. 15 (4), 582–606. http://dx.doi.org/10.1016/j.dt.2019.04.011.

Sánchez-Ibáñez, J.R., Pérez-del Pulgar, C.J., García-Cerezo, A., 2021. Path planning for autonomous mobile robots: A review. Sensors 21 (23), http://dx.doi.org/10.3390/s21237898.

Santiago, R.M.C., De Ocampo, A.L., Ubando, A.T., Bandala, A.A., Dadios, E.P., 2017. Path planning for mobile robots using genetic algorithm and probabilistic roadmap. In: IEEE 9th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management. HNICEM, pp. 1–5. http://dx.doi.org/10.1109/HNICEM.2017.8269498.

Sfeir, J., Saad, M., Saliah-Hassane, H., 2011. An improved artificial potential field approach to real-time mobile robot path planning in an unknown environment. In: IEEE International Symposium on Robotic and Sensors Environments. ROSE, pp. 208–213. http://dx.doi.org/10.1109/ROSE.2011.6058518.

Sierra-Garcia, J.E., Santos, M., 2024. Combining reinforcement learning and conventional control to improve automatic guided vehicles tracking of complex trajectories. Expert Syst. 41 (2), e13076.

Toda, Y., Kubota, N., 2011. Path planning using multi-resolution map for a mobile robot. In: SICE Annual Conference, Vol. 2011. pp. 1276–1281.

Vakaruk, S., Sierra-García, J.E., Mozo, A., Pastor, A., 2021. Forecasting automated guided vehicle malfunctioning with deep learning in a 5g-based industry 4.0 scenario. IEEE Commun. Mag. 59 (11), 102–108.

van der Molen, G.M., 1992. Robotic systems: Advanced techniques and applications. In: Ch. Trajectory Generation for Mobile Robots with Clothoids, vol. 10, Springer Netherlands, Dordrecht, pp. 399–406. http://dx.doi.org/10.1007/978-94-011-2526-0_46.

Wang, Y., Chen, P., Jin, Y., 2009. Trajectory planning for an unmanned ground vehicle group using augmented particle swarm optimization in a dynamic environment. In: 2009 IEEE International Conference on Systems, Man and Cybernetics. pp. 4341–4346. http://dx.doi.org/10.1109/ICSMC.2009.5346947.

Werling, M., Ziegler, J., Kammel, S., Thrun, S., 2010. Optimal trajectory generation for dynamic street scenarios in a frenét frame. In: IEEE International Conference on Robotics and Automation. pp. 987–993. http://dx.doi.org/10.1109/robot.2010.5509799.

Wolter, D., Latecki, L.J., Lakämper, R., Sun, X., 2004. Shape-based robot mapping. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 3238, pp. 439–452. http://dx.doi.org/10.1007/978-3-540-30221-6_33.

Xing, X., Zhao, B., Han, C., Ren, D., Xia, H., 2022. Vehicle motion planning with joint cartesian-frenét mpc. IEEE Robot. Autom. Lett. 7 (4), 10738–10745. http://dx.doi.org/10.1109/LRA.2022.3194330.

Yan, Z., Jouandeau, N., Cherif, A.A., 2013. Acs-prm: Adaptive cross sampling based probabilistic roadmap for multi-robot motion planning. In: Lee, S., Cho, H., Yoon, K.-J., Lee, J. (Eds.), Intelligent Autonomous Systems 12: Volume 1 Proceedings of the 12th International Conference IAS-12, Held June (2012) 26-29, Jeju Island, Korea. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 843–851. http://dx.doi.org/10.1007/978-3-642-33926-4_81.

Yang, S.X., Hu, Y., h. Meng, M.Q., 2006. A knowledge based ga for path planning of multiple mobile robots in dynamic environments. In: IEEE Conference on Robotics, Automation and Mechatronics. pp. 1–6. http://dx.doi.org/10.1109/RAMECH.2006.252703.

Yang, S.X., Meng, M., 2000. An efficient neural network approach to dynamic robot motion planning. Neural Netw. 13 (2), 143–148.

Zamri, Nur Ezlin, Azhar, Siti Aishah, Sidik, Siti Syatirah Muhammad, Mansor, Mohd Asyraf, Kasihmuddin, Mohd Shareduwan Mohd, Pakruddin, Siti Pateema Azeyan, Pauzi, Nurul Atirah, Nawi, Siti Nurhidayah Mat, 2022. Multi-discrete genetic algorithm in hopfield neural network with weighted random k satisfiability. Neural Comput. Appl. 34 (21), 19283–19311. http://dx.doi.org/10.1007/s00521-022-07541-6.

Zhang, Y., wei Gong, D., hua Zhang, J., 2013. Robot path planning in uncertain environment using multi-objective particle swarm optimization. Neurocomputing 103, 172–185. http://dx.doi.org/10.1016/j.neucom.2012.09.019.