DOI: 10.24996/ijs.2024.65.2.38

ISSN: 0067-2904

Path Planning for Autonomous Mobile Robots Using the RFO-GWO Optimization Algorithm

Fetoh H. Ketafa [1]*, Salah Al-Darraji [2]

[1]*Department of Computer Science, College of Computer Science and Information Technology, University of Basrah, Basrah, Iraq*

[2]*Department of Computer Science, College of Education for Pure Sciences, University of Basrah, Basrah, Iraq*

Received: 22/11/2022 Accepted: 19/3/2023 Published: 29/2/2024

Abstract

Path planning is a challenging navigation problem that can be handled using multi-objective methods. This paper presents a three-stage multi-objective path-planning method. The first stage is to locate the best or near-best solution path and avoid detected obstacles using a hybrid of the red fox–gray wolf optimizer (RFO–GWO), which finds a route from the start position to the target position. In the second step, a mutation operation using an evolutionary algorithm is utilized to enhance the length, integrity, and smoothness of the route generated by the RFO–GWO algorithm. The final step of the suggested method is refined further using a multiphase technique. By integrating the real sizes of the mobile robots and the size of the barriers and phrasing the issue as a traveling object in the available area, the suggested path-planning method resembles the actual world. The simulation results indicate that this strategy creates the most viable path even in complicated surroundings, overcoming the disadvantages of traditional approaches. Furthermore, when compared to prior path-planning methods, the simulation's outcomes indicate that the suggested RFO–GWO method is effective in terms of the route, and the strategy is extremely competitive. The results showed a significant improvement, where the total percentage convergence time (in seconds) for RFO–GWO for the three maps was 15%, 12%, and 10%, respectively, whereas it was 35%, 41%, and 43% seconds in GWO and 34%, 35%, and 37% seconds in RFO. There was also a significant improvement in the number of nodes for RFO-GWO (2%, 3%, and 2%) compared to GWO nodes (64%, 65%, and 62%), and RFO nodes (32%, 30%, and 35%) for the same three maps. Subsequently, the smoothness of the path formed by the recommended approach was enhanced using the evolutionary algorithm (EA), where the total percentage length of the path in the worst scenario for GWO was 28% and for RFO was 26% in units, but after improvement with the RFO-GWO with EA, it became 22% in units.

Keywords: Red Fox Optimization Algorithm, Gray Wolf Optimizer, Path Planning, Mobile Robots, Bio-inspired.

*Email: itpg.fatoh.hason@uobasrah.edu.iq

تخطيط المسار للروبوت المحمول المستقل باستعمال خوارزمية الثعلب الأحمر المحسنة و الذئب الرمادي

فتوح حسون كطافة[1]*, صلاح الدراجي[2]

[1] قسم علوم الحاسوب، كلية علوم الحاسوب وتكنولوجيا المعلومات، جامعة البصرة، البصرة، العراق

[2] قسم علوم الحاسوب، كلية التربية للعلوم الصرفة، جامعة البصرة، البصرة، العراق

الخلاصة

يعد تخطيط المسار مشكلة ملاحية صعب التعامل معها باستعمال طرق متعددة الأهداف. تقدم هذه الورقة طريقة تخطيط مسار متعددة الأهداف من ثلاث مراحل. تتمثل المرحلة الأولى في تحديد أفضل أو أقرب مسار للحل وتجنب العوائق المكتشفة باستعمال مزيج من مُحسِّن الثعلب الأحمر – الذئب الرمادي (RFO – GWO)، والذي يجد طريقًا من موضع البداية إلى موضع الهدف. في الخطوة الثانية، يتم استعمال عملية طفرة بخوارزمية تطورية لتعزيز طول وسلامة وسلاسة المسار الذي تم إنشاؤه بواسطة خوارزمية RFO–GWO. تم تنقيح الخطوة الأخيرة من الطريقة المقترحة بشكل أكبر باستعمال تقنية متعددة الأطوار. من خلال دمج الأحجام الحقيقية للروبوتات المتنقلة وحجم الحواجز وصياغة المشكلة ككائن متنقل في المنطقة المتاحة، فإن طريقة تخطيط المسار المقترحة تشبه العالم الفعلي. تشير نتائج المحاكاة إلى أن هذه الإستراتيجية تخلق أفضل مسار قابل للتطبيق حتى في محيط معقد، متغلبًا على عيوب الأساليب التقليدية. علاوة على ذلك، عند مقارنتها بأساليب تخطيط المسار السابقة، تشير نتائج المحاكاة إلى أن طريقة RFO–GWO المقترحة فعالة من حيث المسار، والاستراتيجية تنافسية للغاية، حيث أظهرت النتائج تحسنًا كبيرًا، وتقارب النسبة المئوية الإجمالية وكان وقت RFO–GWO للخرائط الثلاث (15٪، 12٪، و 10٪) ثوانٍ على التوالي، بينما كان (35٪ ، 41٪ ، 43٪) بالثواني في GWO ، وكان (34٪) و 35٪ و 37٪) في ثوانٍ في RFO. كان هناك أيضًا تحسن كبير في عدد العقد لـ RFO–GWO كان (2٪، 3٪، و 2٪) مقارنة بعقد 64) GWO٪، 65٪، و 62٪)، وعقد 32) RFO٪) و 30٪ و 35٪) لنفس الخرائط الثلاث. بعد ذلك، تم تحسين نعومة المسار الذي شكله النهج المقترح باستعمال الخوارزمية التطورية (EA)، حيث كانت النسبة المئوية الإجمالية لطول المسار في أسوأ سيناريو لـ 28) GWO٪) وبالنسبة لـ RFO كانت (26٪) في الوحدات، ولكن بعد التحسين مع RFO–GWO مع EA ، أصبحت (22 ٪) في الوحدات.

1. Introduction

Scientists have recently focused their attention on the development of robotics using artificial intelligence to achieve the autonomy of mobile robots. Autonomous mobile robots can be seen in many fields such as space, industry, transportation, and definition, as well as other social areas, and their use is increasing day by day [1]. Motion planning became autonomous and programmable as digital electronics and computer technology advanced, as did their compatibility using artificial intelligence (AI) methodologies [2]. Mobile robot navigation is mostly determined by its intelligence. Path planning is the most efficient and crucially intelligent component of these capabilities. Path planning is the process of determining the optimum, collision-free route from one location to another. Based on the nature of the environment, it may be classified into two types: static path planning, in which barriers do not change position over time, and dynamic path planning, in which the location and orientation of obstacles change over time. [3], [4].

These are further divided into offline and online methods based on the mobile robot's information level. The mobile robots have entire information about the situation when offline path planning. Over the past four decades, mobile robot navigation has become a growing field. Several traditional and AI strategies have been used to address this issue [5]–[7]. A mobile robot's path planning is divided into three major tasks. This includes understanding the environment, determining one's current location, and finally taking the appropriate steps and incorporating the gathered information to complete tasks [8], [9].

The primary challenges for autonomous mobile robot path planning are complexity, flexibility, and effective path choosing, which assures effective and collision-free paths by bypassing scenarios such as dead ends. LaValle [10] illustrates a compelling reason to discuss path planning methods. It is an intriguing subject for machines to handle a variety of tasks that humans typically find difficult to solve. This also necessitates the creation and development of a methodical and strong algorithm for path planning and achievement.

Navigation research dates back to the 1960s, and several techniques, including cell decomposition [11], roadmap techniques [12], and the potential field [13], have been proposed. The major weaknesses of these techniques are their ineffectiveness because of high operational costs and their inexactness as a result of the increased risk of being caught in a local optimal solution. Implementing different heuristic techniques, such as bio-inspired neural systems and genetic methods, can solve the drawbacks of these algorithms [14]. Rapidly satisfying answers are among the most important focuses of these heuristic techniques, which are especially suitable for solving NP-complete problems.

The contribution of this research is the development of a new path-planning method in three phases.

1. The A phase creates an obstacle-free path using a combination of two inspired algorithms: the RFO and GWO. A path is produced to achieve the goal of the multi-objective tasks described in this paper.
2. The B phase adds mutation operations via an evolutionary algorithm (EA) to make a path semi-smooth.
3. In the C phase, the RFO–GWO algorithm is enhanced by a multiphase technique that transforms the resulting path of sharp zigzags into a smoother path due to the reduction of zigzags.

The rest of this paper is organized as follows: Section 2 highlights numerous research approaches. Section 3 introduces population-based optimization as well as the approaches developed in this work for mobile robot path planning. Section 4 describes the proposed method. Section 5 presents a series of simulation findings to demonstrate the usefulness of the suggested technique compared to earlier efforts. Section 6 discusses the findings, and Section 7 contains the conclusions and recommendations.

2. Related works

There are three sorts of existing path planning techniques: classical algorithms, heuristic algorithms, and meta-heuristic algorithms [14]. The road map technique (RM), cell decomposition technique (CD), rapidly exploring random tree (RRT) technique, and potential field technique (PF) are examples of classical methodologies. Heuristic approaches include the D algorithm, Dijkstra's algorithm, and the A* search algorithm. algorithm (GA), an artificial neural network (ANN), whose modified forms are commonly employed to discover the optimal length of path for mobile robot path planning in several situations [15], and Particle Swarm Optimization (PSO) are examples of evolutionary algorithms. All of these algorithms and methods are widely employed for mobile robot navigation. The research in [16]–[18] used particle swarm optimization (PSO) for mobile robot navigation from the start position to the target position while bypassing barriers on the robot's route.

Numerous meta-heuristic techniques, such as nature-inspired algorithms, have also been employed to handle multi-objective navigation issues for mobile robots. Several prior studies have used examples of natural behaviors in this group. The research in [19]–[22] used Ant Colony Optimization (ACO) to address path planning issues in complicated settings. In [23], an enhanced version of ACO (IACO) is presented to achieve quicker convergence times and avoid trapping in a local optimum. The IACO gave the best path when compared with other algorithms, nevertheless; it has a low convergence speed.

Several publications have also used bio-inspired methodologies to address different aspects of path planning strategies. [24]–[31], a Whale Optimization Algorithm (WOA), applied in fixed situations to meet prerequisites for the optimization length of path and smoothing path [32], and in [33], [34], proposed a technique that relies on the Cuckoo Optimization Algorithm for planning the robot's path in a moving situation. The simulation findings indicate that the method finds a barrier-free and short path under a variety of environmental situations.

In [35], a proposal for an obstacle avoidance Robot is an autonomous robotic vehicle that notices barriers on its route via its detectors, avoids them, and draws a conclusion that relies on the internal code put into it.

To improve robot route optimization methods, hybridized meta-heuristic methods were also used. The goal of combining numerous metaheuristic approaches is to integrate their benefits to create a superior algorithm. The genetic algorithm with particle swarm optimization (GA-PSO) and Multi-Objective Bare Bones Using Particle Swarm Optimization with Differential Evolution are two hybridized ways for designing a path for a robot (MOBBPSO) [36], [37]. Previous studies had the constraint of seeing the mobile robot as a single particle. Some of these strategies were created in order to find the shortest route while avoiding immovable obstacles. Other research focuses on avoiding moving barriers while choosing the quickest path without taking the road's smoothness into account. Furthermore, despite the simplicity with which some of the prior research's grid-based techniques might be implemented, they have various shortcomings, including the imperfect performance of the obstruction, where even if the barrier covers just a tiny part of the cell, the full cell is designated for that barrier. In dynamic contexts, this wastes space and hinders adaptability.
Faiza et al. [38] suggested a strategy based on a hybridized Gray Wolf optimizer with the particle swarm optimization method. The approach was based on an obstacle detection and avoidance method. The strategy used evolutionary mutation operations to tackle path integrity and smooth it down even more for an autonomous mobile robot. Several experiments were run in various situations to verify the probability of the suggested approach, and it was discovered that the method provides more viable paths with shorter distances.

Połap and Woźniak [39] presented the Red Fox Optimizer (RFO), a novel metaheuristic optimization technique. The RFO algorithm is dependent on red fox population behavior and inspired by red fox natural behaviors, such as foraging, hunting, and population growth while escaping hunters. This approach has lately been utilized to tackle a large number of optimization issues [40], [41]. The RFO method was used to solve the path planning problem of the autonomous mobile robot in [42]. The significance of this method is in using RFO to solve the navigational problems of autonomous mobile robots using polynomial logistic regression to reduce the fluctuations of the resulting path and produce a semi-smooth path.
RFO was integrated with deep long- and short-term memory (LSTM) in a neural network (deep LSTM-RFO) technique for DR classification. Priya et al. [43] proposed integrating and

employing RFO to improve the performance of deep LSTM during classification. Polap et al. [44] developed a federated learning hybrid combining artificial intelligence training and a meta-heuristic, with the Red Fox Optimization algorithm acting as a representation of the meta-heuristic.

The Gray Wolf Optimization (GWO) algorithm, introduced by Mirjalili in 2014, mimics gray wolf hunting tactics and social leadership [45]. For tackling robot route planning difficulties, the GWO method [46] was suggested Click or tap here to enter text.to discover the best route from the start location to the goal location while bypassing barriers. In this work, the gray wolf optimization (GWO) approach is utilized as an obstacle avoidance instrument, in which an RFO is connected to it to implement target-seeking behavior, and evolutionary mutation operators are added to address route integrity and smooth it more for an autonomous mobile robot.

3. Population-based optimization

Meta-heuristic algorithms have proved excellent at dealing with optimization problems involving complicated rules, and they have been commonly used in several mobile robot path planning challenges. In a decentralized self-organizing system, population-based meta-heuristic (PMH) algorithms demonstrate collectively intelligent behavior. The system is made up of numerous individuals, and intelligent behavior is shown through interactions between homogenous individuals or through the environment. There is no main controller in the entire population, and individuals update themselves by communicating according to basic norms; therefore, this interaction leads to global-level intelligence. There are many instances of intelligent behavior in nature, including ants, birds, and fish, and several common PMH algorithms are utilized in mobile robot path planning situations as they are all population-based and repetition-based algorithms that are inspired by nature or society. Their distinctions lie not only in the process of behavior inspiration but also in the method of exploitation or exploration. This study employs two such algorithms.

3.1 Red fox optimization algorithm

Dawid Poap and Marcin Woniaki proposed the red fox algorithm for optimization purposes in 2021 [39]. The RFO is a mathematical depiction of red fox behavior that includes foraging, hunting, and population growth while fleeing from hunters. The model employs a replication mechanism and local and global optimization techniques. The RFO algorithm is a brand-new meta-heuristic optimization tool inspired by red fox hunting methods. Individuals with well-defined territories and nomads make up the red fox population. According to the alpha couple’s structure, each herd shares a specific territory, but if young animals obtain possession of another region, they may quit the herd; otherwise, the hunting area is passed from parents to their children. When the fox discovers that its prey is close, its territorial hunt is depicted as a global search. The fox wanders across its environment to get as close to the victim as possible before staging an assault. A local search is shown as a territorial traversal to get as close to the prey as possible before attacking it, so the RFO algorithm replicates a solution area reconnaissance to perform a worldwide search when the fox detects nearby prey while foraging.

The RFO algorithm, like other meta-heuristics, comprises exploitation and exploration. Exploration is characterized by the fox’s choice of prey in faraway regions, whereas exploitation is defined by the fox's desire to get close to the prey whenever it is feasible to attack it. The following describes RFO initialization:

In each cycle, the population of individuals includes a set of foxes that is described by a point $\bar{x} = [x_0, x_1, \ldots, x_{n-1}]$ with $\boldsymbol{n}$ coordinates. To distinguish a fox x^i in iteration $\boldsymbol{t}$, use the notation $(x_j^i)^t$, where $\boldsymbol{i}$ is the fox's number in the population and $\boldsymbol{j}$ is the coordinate depending on the size of the solution space. To achieve the best values for the criterion function, assume that foxes roam across the solution region using the supplied equations.
Each fox in a herd must perform a critical role for the entire family to survive. Individuals in the herd will migrate to distant places to seek new territory if the habitat is empty of food. They share the knowledge they gain from this investigation with their family to help them survive and thrive. It is forecast that each individual will explore the surrounding locations based on their overall fitness, and the recommended strategy assumes that the best individual travels to the most fascinating areas and teaches this knowledge to their family. As a consequence, the population is sorted first by the fitness term, then the squares of each individual's Euclidean distance are computed $(\bar{x}^{best})^t$, and population members are pointed toward the best individual as follows:

$$d((\bar{x}^i)^t, (\bar{x}^{best})^t) = \sqrt{\|(\bar{x}^i)^t - (\bar{x}^{best})^t\|}, \tag{1}$$

and point population members toward the best individual.

$$(\bar{x}^i)^t = (\bar{x}^i)^t + \alpha sign((\bar{x}^{best})^t - (\bar{x}^i)^t), \tag{2}$$

where $\alpha \in (0, \mathrm{d}((\bar{x}^i)^t, (\bar{x}^{best})^t)\rangle$ represents a single, iterative adjustment of a randomly chosen scaling hyperparameter for the whole population. Individuals stay in the optimal location if the fitness of the most recent location is the best; otherwise, they return to their previous place. These operations explain the planned global search performed for each RFO iteration.

3.2 Gray wolf optimizer algorithm

The GWO algorithm was suggested by Mir Jalili in 2014. It mimics the hunting technique and communal management of gray wolves [45]. Gray wolves are classified into four tiers in this algorithm based on their social hierarchy: alpha, beta, delta, and omega. An alpha wolf is the pack leader, and omega wolves are gray wolves at the lowest level. Scouts, guards, elders, hunters, and caretakers make up this group. The gray wolf hunting strategy is an intriguing method within the GWO algorithm, in addition to the social leader mechanism.

Another fascinating social activity of gray wolves is that they hunt in groups. First, the gray wolves locate the prey and circle around it under the command of the alpha wolf. The mathematical model of the gray wolf hunting strategy assumes that alpha, beta, and delta wolves provide greater information about prospective foraging positions. As a result, the top three optimal solutions (alpha, beta, and delta) are used to change the wolves' placements in the GWO algorithm. There are no omega wolves in the GWO code [45]. The following is an arithmetic model of the gray wolf hunting technique:

$$\vec{D}_\alpha = \left|\vec{C}_\alpha \,.\, \vec{X}_\alpha - \vec{X}_i\right| \tag{3}$$

$$\vec{D}_\beta = \left|\vec{C}_\beta \,.\, \vec{X}_\beta - \vec{X}_i\right| \tag{4}$$

$$\vec{D}_\delta = \left|\vec{C}_\delta \,.\, \vec{X}_\delta - \vec{X}_i\right| \tag{5}$$

$$\vec{U}_\alpha = \vec{X}_\alpha - \vec{A}_\alpha \vec{D}_\alpha \tag{6}$$

$$\vec{U}_\beta = \vec{X}_\beta - \vec{A}_\beta \vec{D}_\beta \tag{7}$$

$$\vec{U}_\delta = \vec{X}_\delta - \vec{A}_\delta \vec{D}_\delta \tag{8}$$

$$\vec{X}_i = \left(\vec{U}_\alpha + \vec{U}_\beta + \vec{U}_\delta\right)\Big/3 \tag{9}$$

where $\vec{D}_\alpha$, $\vec{D}_\beta, \vec{D}_\delta$ denotes the distance vector between the prey and the wolf (alpha, beta, delta), $\vec{X}_\alpha$, $\vec{X}_\beta, \vec{X}_\delta$ denotes the location vector of the prey, $\vec{X}_i$ denotes the position vector of the gray wolf at i_{th} iteration, $\vec{C}_\alpha$, $\vec{C}_\beta, \vec{C}_\delta$ $\vec{A}_\alpha$, $\vec{A}_{\beta,}\vec{A}_\delta$ denotes the coefficient vectors of alpha, beta, and delta wolves, and $\vec{U}_\alpha$, $\vec{U}_\beta, \vec{U}_\delta$ denotes the trial vector for the alpha, beta, and delta wolves. The coefficient vectors for the alpha, beta, and delta wolves are as follows:

$$\vec{A}_\alpha = 2\vec{a}\vec{r}_{\alpha 1} - \vec{a} \tag{10}$$
$$\vec{C}_\alpha = 2\vec{r}_{\alpha 2} \tag{11}$$
$$\vec{A}_\beta = 2\vec{a}\vec{r}_{\beta 1} - \vec{a} \tag{12}$$
$$\vec{C}_\beta = 2\vec{r}_{\beta 2} \tag{13}$$
$$\vec{A}_\delta = 2\vec{a}\vec{r}_{\delta 1} - \vec{a} \tag{14}$$
$$\vec{C}_\delta = 2\vec{r}_{\delta 2} \tag{15}$$

where $\vec{a}$ denotes the vector that was linearly dropped from 2 to 0 throughout the optimization, $\vec{r}_{\alpha 1}$, $\vec{r}_{\beta 1}, \vec{r}_{\delta 1}$denotes the first random vector in [0,1], and $\vec{r}_{\alpha 2}$, $\vec{r}_{\beta 2}, \vec{r}_{\delta 2}$denotes the second random vector in [0,1]. Figure 1 depicts the hunting process of the gray wolf group, where group members adjust their places based on the alpha, beta, and delta wolves and prey. The gray wolves take their victim and end the hunt by attacking it. This condition is described as a decreasing $\vec{a}$ vector in the mathematical model shown below:

$$\vec{a} = 2 - \frac{2\,.Iter}{MaxIt} \tag{16}$$

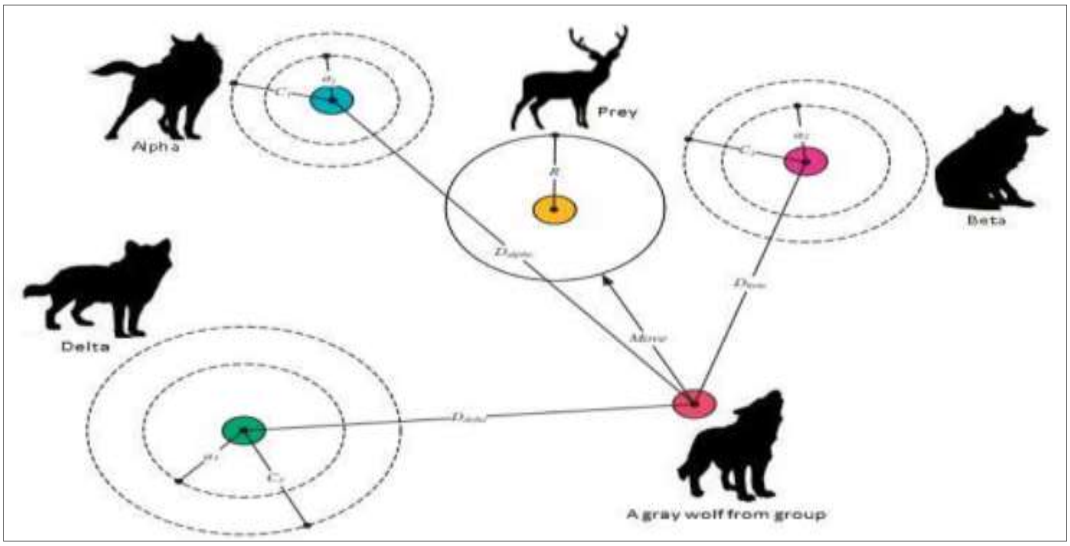


Figure 1: The hunting technique of gray wolves [46]

4. Proposed method

The suggested method for autonomous mobile robot path planning is described in this section and is based on hybridized population optimization and path-smoothing techniques.

4.1 Hybrid RFO–GWO algorithm

To improve overall effectiveness, the best aspects of several optimization methods are blended to generate a hybrid optimization method. The major task here is to create a suitable path that will assist the robot in determining its path from the beginning point to the target location. The proposed research technique is a mixture or hybridization of RFO [39] and

GWO [45]. The path-planning issue is recast as a minimization problem. The RFO algorithm takes the shortest route from the fox to the target. The GWO algorithm allows the fox to move around its surroundings while effectively avoiding obstacles. Consider the robot environment in Figure 2, which clearly depicts the robot's beginning position and objective point. The black line represents the robot's path to the goal. While the fox moves around the environment, if the point generated by reallocating equation (2) in the RFO algorithm generates a line segment that connects to a previous node and goes through a barrier, the node is considered infeasible, or if the place is smaller than one meter from the border of any barriers, the fox should avoid moving there, and the GWO algorithm initializes around the fox's current location to bypass the barriers on the path. The GWO algorithm will determine the ideal next location for the robot to move to avoid colliding with barriers on the way to its destination. Each step taken by the robot is determined by the distance between its optimum location and the goal and the barriers in its environment.

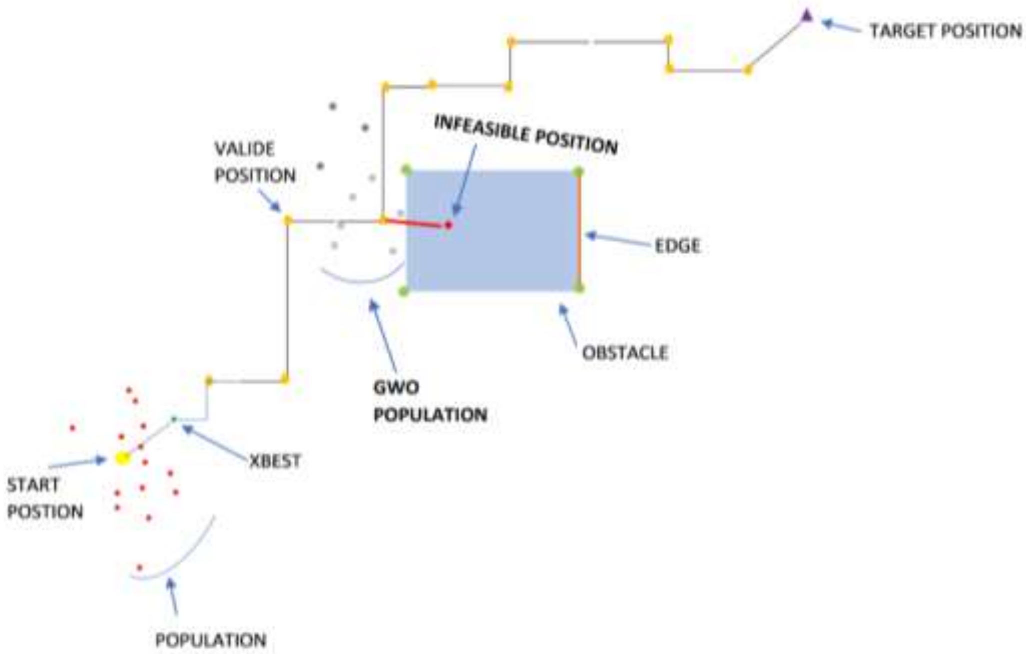


Figure 2: RFO-GWO path planning

Initially, the RFO algorithm is initialized by generating a population around the starting point. The population is then evaluated by the fitness function, and a definition of the fitness function is necessary. This is computed for every individual in the population, where the fitness function is determined by the Euclidean distance between each member and the target location according to the following equation:

$$d((\bar{x}^i)^t, target) = \sqrt{\|(\bar{x}^i)^t - target\|}$$

The best individual is selected depending on the values of the fitness function. The individual with the lowest value is the greatest member of the population, as the fitness function is a minimization function. Then, valid positions found by the RFO–GWO algorithm are added to an empty list called the location history. The first operation is to move the fox to the new valid position using the reallocation equation (2) that determines a new location for the fox. A new location toward the objective is produced at random depending on the location of the best individual in the population. To reduce the time spent on reaching the goal, the distance between the current location of the individual and the location of the goal relies on choosing the random value α where $\alpha \in (0, \mathrm{d}((\bar{x}^{i})^{t}, (\bar{x}^{target})^{t}))$. If the new location is greater than the previous location, i.e., closer to the target, and the new location generates a line segment that connects to a former node but does not pass an obstruction, or if this location is more than 20 cm from the border of any barriers, it is safe for the fox to step toward it. This node is considered executable and is added to the location history list; otherwise, the new position is infeasible.

The GWO algorithm is initialized around the current location of the fox to bypass the barrier in the path, as shown in algorithm (1). The current location of the fox and the current obstacle are given to the GWO algorithm. The wolf's population is generated around the current location of the fox. The population is evaluated by calculating the distance between each member and the current obstacle. The fitness function is a maximization function, where the furthest from the obstacle is the best. The three best locations, i.e., the three locations furthest from the obstacle, are chosen as alpha, beta, and delta. The a vector is calculated for each GWO iteration, and a coefficient vector is generated for each individual by equations 10–15. The distance vector and trial vectors are calculated by equations (3–5) and (6–8), respectively; then, the new location that is furthest from the obstacle is calculated based on equation (9). If the wolf's new location is better than the fox's current location, the new location is valid. The GWO algorithm returns the new location to the RFO algorithm, so the fox moves to this location, which is then added to the location history list; otherwise, the fox's current location is returned, meaning that the fox will remain in its current location for this iteration, as shown in algorithm (2).

Algorithm 1: Hybrid RFO-GWO path planning algorithm

Input: start_ position, target_position, obstacles
Output: path
Setup: T is maximum iteration; n is the size of population.
Generate population consisting of n foxes around start position
Create empty list named Location_history for each fox
$t := 0$
Define scaling parameter α
while $t \leq T$ **do**
Sort population based on the fitness function Eq. (17)
Select $(xbest)^{t}$
foreach fox_i in the current population **do**
Calculate the new position of individuals according to Eq. (2)
if the new location is valid **then**
if the new position is better than previous position
then
add new_position to Location_history list of fox_i
Current_position = new_position
else
Calculate the new_position of individuals by GWO
add new_position to Location_history list of fox_i
Current_position = new_position
endif
else
Calculate the new_position of individuals by GWO
add new_position to Location_history list of fox_i

```
        Current_position = new_position
    endif
    if current_position = target_position then
      return the Location_history list
    endif
    end for
    Update fitness of each fox
    t++
    end while
Stop
```

Algorithm 2: GWO obstacle avoidance algorithm

```
Input: current position of fox_i, obstacle_j
Output: path
Setup:
    T is maximum iteration, C is coefficient vectors, D is distance vectors.
  Generate population consisting of m gray wolves around current position of fox_i
  Determine the cost of gray wolves depend on distance between current location and obstacle_j
  Keep the better three gray wolf that have maximum cost as alpha, beta, and delta respectively
  t: =0
while (t < T)
  Decrease using Eq. (16)
  foreach wolf
    Create the C for alpha, beta, delta
    Determine the D using Eqs. (3-5)
    Calculate the trial vectors using Eqs. (6-8)
    Calculate the position of gray wolf using Eq. (9)
    If new position is valid then
        if the new location is preferable to the current location of fox_i then
       return new position
      else
       return current position of fox_i
      endif
    else
      return current position of fox_i
    endif
end for
  increase iteration one
end while
Stop
```

4.2 Optimize the path using an evolutionary algorithm

This involves enhancing EA-based motion planning to improve the smoothness of the length of the path. In 1999, Fogel [47] proposed the EA as an improvement to the genetic algorithm (GA) because of its pliability in resolution performance. The GA is concerned with genotypes, whereas the EA is concerned with phenotypic fields. As EA lacks a crossover operation, the evolution method is carried out via mutation operations.

The first population for numerous paths may be built using a valid reference path, i.e., a list of nodes. As illustrated in Figure 3, route P is connected with a list of M vertices: $P = V_1(x,y),\ldots,V_M(x,y)$. The target position of an autonomous mobile robot is indicated by the figure's last vertex. Path planning aims to enhance the most recent route created by the RFO–GWO algorithm.

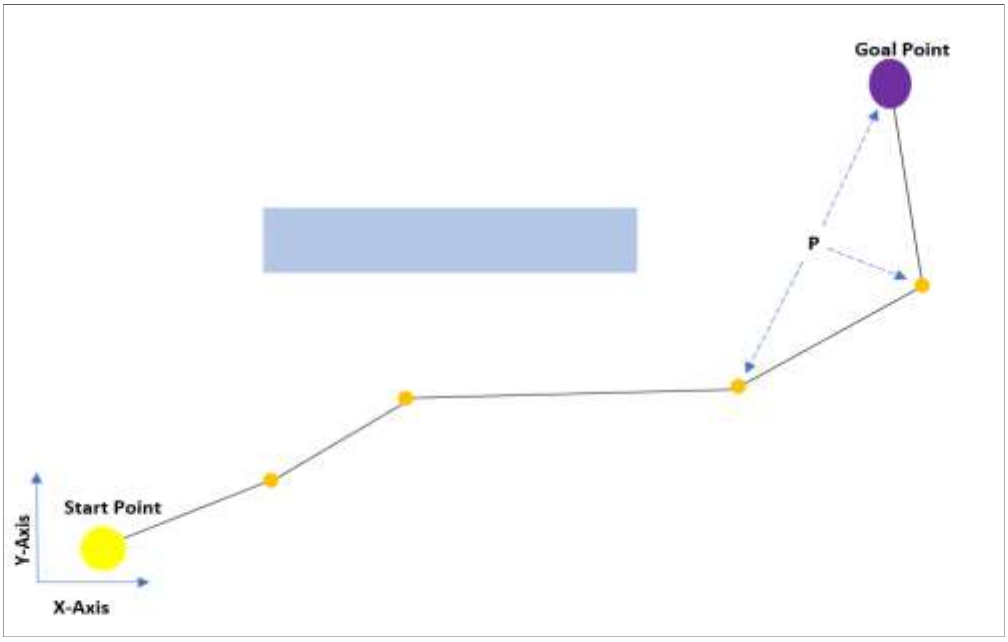


Figure 3: Optimization of path planning to achieve the desired position, a path is connected with diverse vertices

The evolutionary process materializes through mutation. A mutation operator is commonly implemented for an obstacle-free route, and every path within the population develops a new path via the use of mutation operators [7]. This paper introduces a mutation operator called Erase. The work performed by this operator is explained below, and a graphic depiction is presented in Figure 4 [48].

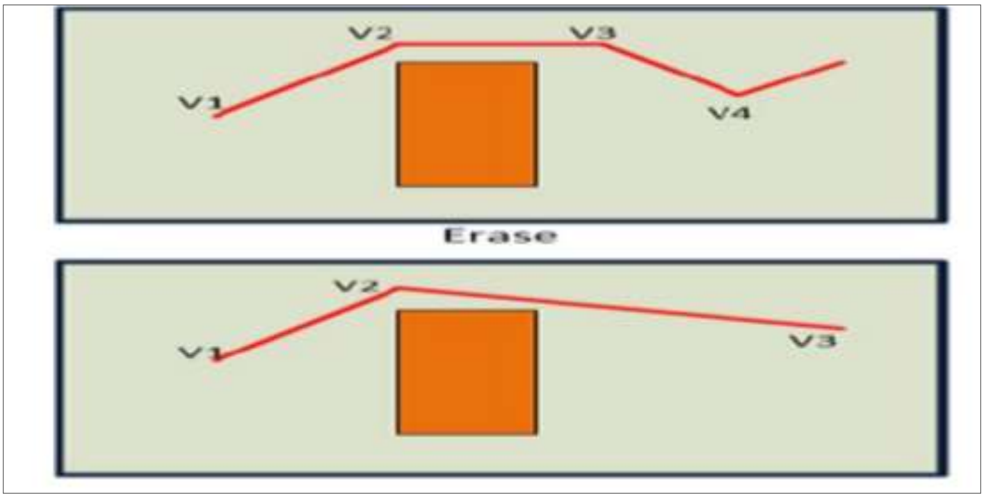


Figure 4: A graphic illustration of the processes of an EA

To refine this further, a multiphase technique is used to find a feasible resolution to the problem of zigzagging on the path by breaking the problem into a sequence of sub-problems that can be solved individually. Initially, the EA is applied along the path. In the next stage, the resulting path is divided into a number of segments that are equal to or less than a specific value. If the distance between points P1 and P2 is greater than the specific value, the P1 and P2 segments are divided into equal segments along the path. In the next step, the EA is applied again.

5. Simulation results

The suggested method was evaluated by comparing the RFO and GWO. Simulations were performed using the Python environment, and every simulation was validated. The trials were simulated on a laptop running Ubuntu 20 and powered by an Intel(R) Core (TM) i7-11800H processor operating at 2.30 GHz with 8 GB of memory. Using three environments crowded with different obstacles, the size of each environment was 17,000 * 17,000 cm. Each algorithm was run 100 times in the same three environments. Table 1 presents a list of the algorithm's parameters that were used in the simulation. Fifteen individuals were initialized as a population within the solution space to perform the search. Table 2 displays the performance results. The path length, number of iterations, number of nodes, and consumption time were the criteria used to analyze performance. In the following paragraphs, the findings from the application of these algorithms will be discussed.

Table 1: Supposed parameter values

Character	Depiction	Value
N	Size of population of RFO	10
t	Iterations	2000
r	Random value	[0,3]
M	population size Of GWO	7
it	Iterations	500

Table 2: Analysis and comparison of the simulated results

Methods	***Map-1***			
	After 100 runs, the average time was taken (sec)	Average cost taken after 100 runs	Average nodes taken after 100 runs	Average path length(unit) for 100 runs
GWO	0.99	210	298	52,908
RFO	0.955	181.65	150	51,698
RFO-GWO	0.43	24	8.54	44,907
MP RFO-GWO-EA	0.43	24	9.81	43,122
Methods	***Map-2***			
	After 100 runs, the average time was taken (sec)	Average cost taken after 100 runs	Average nodes taken after 100 runs	Average path length(unit) for 100 runs
GWO	2.3	162	174	48,548
RFO	1.952	105.49	80	44,513
RFO-GWO	0.67	15	6.77	39,589
MP RFO-GWO-EA	0.67	15	8	38,126
Methods	***Map-3***			
	After 100 runs, the	Average cost	Average nodes	Average path

	average time was taken (sec))	taken after 100 runs	taken after 100 runs	length(unit) for 100 runs
GWO	6.30	216	359	47,976
RFO	5.39	250	200	47,508
RFO-GWO	1.46	18	8.26	40,055
MP RFO-GWO-EA	1.46	18	10.32	38,601

Table 2 provides the simulation findings for the three environments. The time taken by the individual who first reached the goal is shown (i.e., the least amount of time taken to produce a path free of obstacles and oscillations). It also indicates the cost to produce this path or the number of iterations that were consumed to achieve the goal. The cost was clarified in an expression of the number of nodes the path passed through to reach the goal, as the generation of each node consumed a computation cost. As noted in the results, the more complex the environment in terms of obstacles, the greater the time spent to reach the goal. When comparing environment A, which was not crowded with obstacles, with crowded environment C, it was noted that the convergence speed slowed with the crowding of the barriers in the situation. Figures 5 and 6 present an improvement in time and path length after using the proposed method.

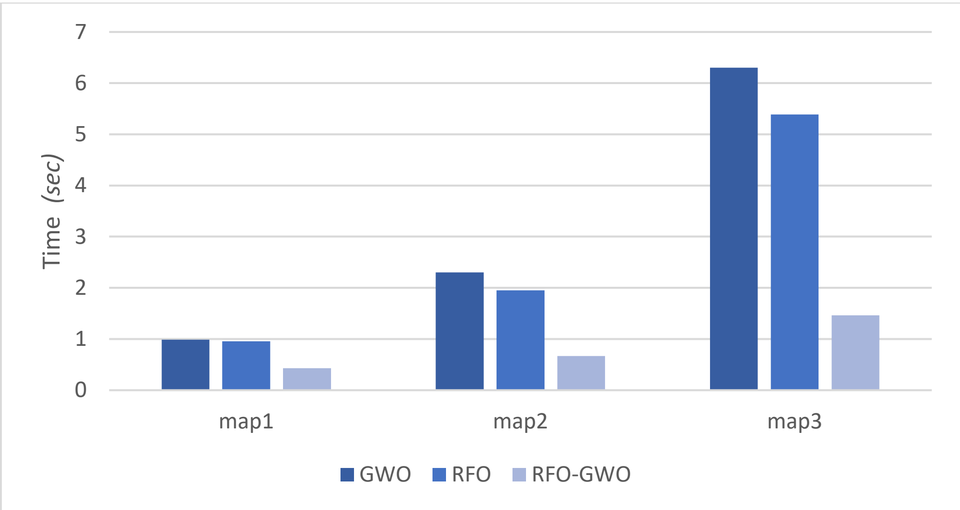


Figure 5: Summary graph of time convergence

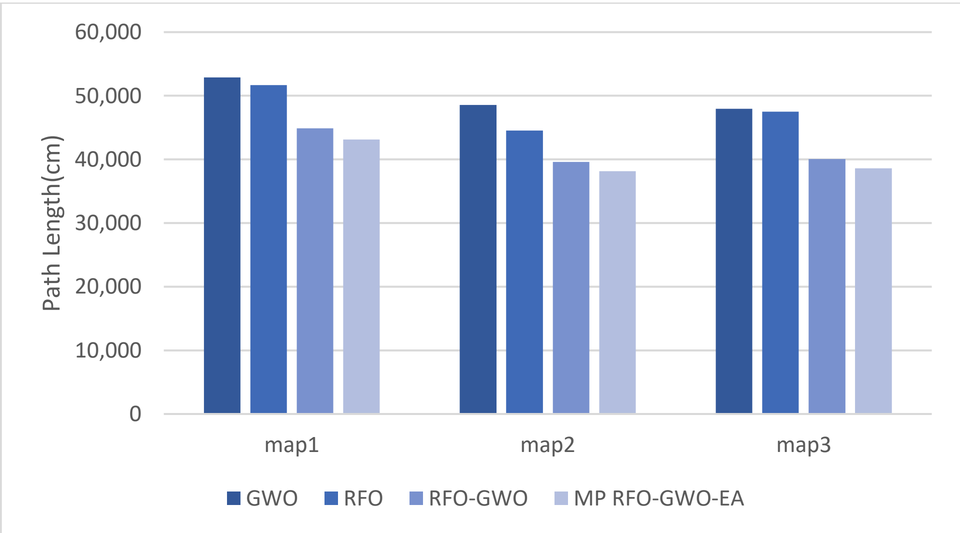


Figure 6: Summary graph of path length

The path generated in environment A was the fastest to produce, as it was not crowded with obstacles, but it also had the longest path. In environment B, more time was taken to generate the path because it was crowded with obstacles, but its path was shorter and had fewer nodes. In other words, it was smoother compared to Environment A. In environment C, which was the most challenging environment for the algorithm because it was heavily crowded with obstacles, it took more time to generate the path, but the length of the path was shorter than in environment A. The results were presented after applying the multiphase technique and EA to the RFO–GWO algorithm. It was noticed that the number of nodes decreased, the path became shorter, and a smoother path was produced.

The GWO and RFO algorithms were applied to the same three maps to demonstrate the performance of the RFO–GWO algorithm. The resulting paths from the algorithms were compared in the three environments, and the RFO–GWO algorithm exceeded the GWO algorithm in the expressions of time, path length, and cost before the path was optimized using the multiphase technique and EA. After applying the multiphase technique and EA, the results were also better in terms of the number of nodes and path length.

Figure 6 compares the obtained results where the robot must begin to travel from the bottom-right corner to the destination area through the barrier situations shown in Figure 7. The findings reveal that the robot gets to the objective without hitting a wall (obstacle). In comparison, the path found using the multiphase RFO–GWO algorithm and EA approach is smoother, as seen in Figure 8.

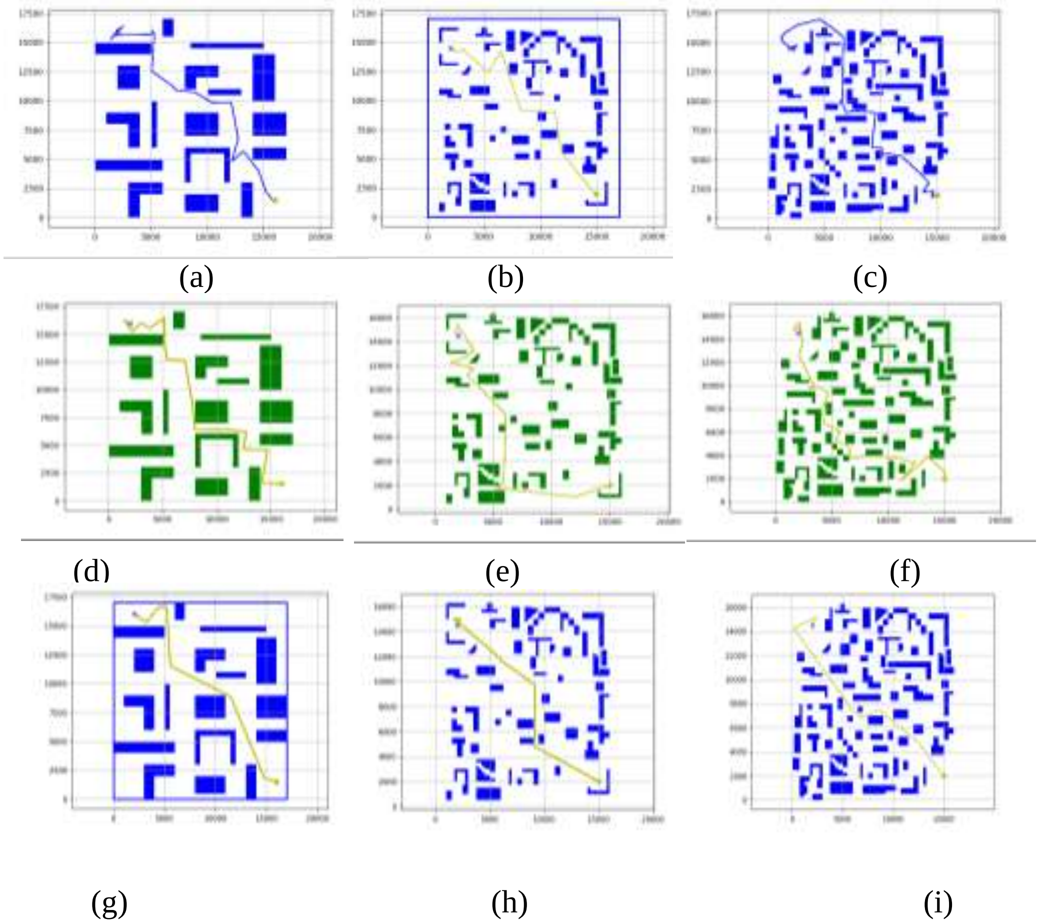
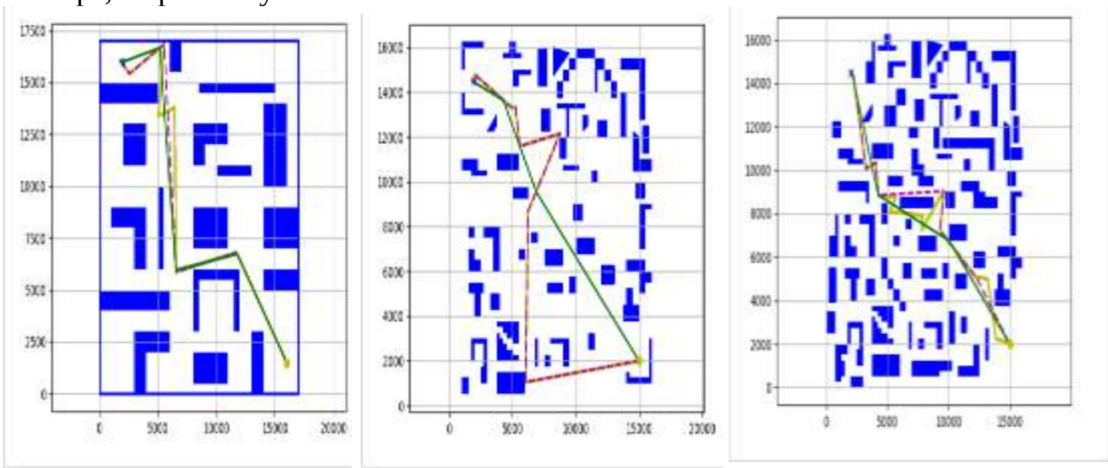


Figure 7: (a–c) RFO algorithm path planning for maps (1–3); (d–f) GWO algorithm path planning for maps (1–3); (g–i) path resulting from the hybrid RFO–GWO algorithm for the three maps, respectively



(a) (b) (c)

Figure 8: Comparison of the three maps between, before, and after using multiphase and the EA, where the yellow line is the original path of the RFO–GWO algorithm, the dashed line is the path after applying EA, and the green line is the optimized path obtained from the proposed multiphase RFO–GWO EA method

Notice that when comparing the results of the RFO–GWO algorithm with the RFO and GWO algorithms, the length of the path and the time spent producing the path became significantly less compared to the two algorithms, and the number of iterations and nodes decreased significantly. However, when comparing the RFO–GWO algorithm with the multiphase RFO–GWO EA method, the path length was reduced.

6. Discussion

According to the simulation findings, the suggested multiphase RFO–GWO EA approach outperforms existing optimization strategies in terms of finding the shortest path. In particular, the combination RFO–GWO algorithm surpasses the RFO in terms of performance standards, as every individual in the RFO calls the GWO when they encounter an obstacle, which reduces the burden on the RFO. Compared to other path-planning approaches in the field, the suggested multiphase RFO–GWO EA method can produce more improvement.
Table 3 compares our suggested technique to the previously discussed methods based on three parameters (cost, time, and path length).

Table 3: Compare Between Methods

Author	Method	Low Cost	High Convergence	Optimal Path
T. F. Abaas and A. H. Shabeeb 2020	PSO [18]	No	Yes	Yes
G. Chen and J. Liu 2019	ACO [21], [22]	Yes	No	Yes
S. Hosseininejad and C. Dadkhah 2019	COA [34]	Yes	NO	Yes
Wang et al. 2016	GA-PSO [37]	Yes	NO	Yes
Zhang et al. 2018	MOBBPSO [36]	NO	NO	Yes
F. Hason and S. Al-Darraji 2022	RFO [42]	NO	Yes	Yes

L. Doğan and U. Yüzgeç 2018	GWO [46]	NO	Yes	NO
Current work	EA RFO-GWO	Yes	Yes	Yes

7. Conclusion and Future Work

In this research, a hybrid strategy for finding the optimal or semi-optimal solution to the path planning issue of the autonomous mobile robot was proposed. This method was applied on three maps in order to evaluate the technique in terms of cost and convergence time. To improve overall effectiveness, the best aspects of several optimization methods are blended to generate a hybrid optimization method. A hybrid method called Red Fox-Gray Wolfe Optimizer (RFO-GWO) has been proposed to improve convergence rate along with cost improvement (in terms of the number of nodes and the number of iterations) and path length improvement.

RFO was combined with GWO to take advantage of the advantages of both methods to go towards the goal with a low convergence time and low cost, where the improvement in the results was very noticeable, as the total percentage convergence time for RFO-GWO for the three maps was 15%, 12%, and 10% seconds, respectively, as well as the total percentage number of nodes for RFO-GWO, as it had 2%, 3%, and 2% nodes for the same three maps, and there is also a noticeable improvement in the length of the path. Then the smoothness of the path generated by the suggested method was improved using the evolutionary algorithm (EA), where the total percentage length of the path in the suggested method RFO-GWO with EA became 22% in units.

The simulation findings showed that the suggested hybridization EA RFO-GWO method demonstrated its usefulness in creating an ideal semi-smooth path by reducing the number of nodes and successfully navigating to its objective by avoiding static barriers in a straightforward and time-saving manner while maintaining the trajectory's smoothness and the efficiency of the path, and also outperformed current state-of-the-art solutions. Despite the proposed approach's being proven to be adaptable and the efficiency of path planning, reinforcement learning produces superior results. Additionally, the effectiveness of the suggested approach is exclusively evaluated using the path planning of a single mobile robot's path around fixed barriers. A viable area for future study is to apply the suggested path planning method in an elaborate, dynamic situation with several mobile robots and a moveable target.

References

[1] L. A. Al-Ani, M. Sahib, M. Altaei, and Q. Khamees, “Simulation of Obstacle Avoidance for Auto Guided Land Vehicle,” *Iraqi Journal of Science*, vol. 60, no. 1, pp. 189–203, Jan. 2019, Doi: 10.24996/ijs.2019.60.1.20.

[2] M. N. Zafar, J. C. Mohanta, and A. Keshari, “GWO-Potential Field Method for Mobile Robot Path Planning and Navigation Control,” *Arab J Sci Eng*, vol. 46, no. 8, pp. 8087–8104, Aug. 2021, Doi: 10.1007/S13369-021-05487-W.

[3] L. B. Amar and W. M. Jasim, “Hybrid metaheuristic approach for robot path planning in dynamic environment,” *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 4, pp. 2152–2162, Aug. 2021, Doi: 10.11591/EEI.V10I4.2836.

[4] M. Sood and V. K. Panchal, “Meta-heuristic techniques for path planning: Recent trends and advancements,” *International Journal of Intelligent Systems Technologies and Applications*, vol. 19, no. 1, pp. 36–77, 2020, Doi: 10.1504/IJISTA.2020.105177.

[5] H. Henrich, D., Wurll, C., & Worn, “On-line path planning by heuristic hierarchical search,” *Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (Cat. No. 98CH36200). 1998. IEEE.*, 1998.

[6] P. R. B. Wehn, H.W And, "Ultrasound-based robot position estimation," *IEEE transactions on Robotics and Automation*, Vol. 13, No. 5, pp. 682-*692*, 1997.
[7] J. M. ,Zhang, L. Trojanowski, K. Xiao, "Adaptive evolutionary planner/navigator for mobile robots," *IEEE transactions on evolutionary computation*, Vol. 1, no. 1, pp. 18-28, 1997.
[8] B. B. V. L. Deepak, D. R. Parhi, and B. M. V. A. Raju, "Advance Particle Swarm Optimization-Based Navigational Controller For Mobile Robot," *Arab J Sci Eng*, vol. 39, no. 8, pp. 6477–6487, 2014, Doi: 10.1007/S13369-014-1154-Z.
[9] K. H. Sedighi, K. Ashenayi, T. W. Manikas, R. L. Wainwright, and H. M. Tai, "Autonomous local path planning for a mobile robot using a genetic algorithm," *Proceedings of the 2004 Congress on Evolutionary Computation, CEC2004*, vol. 2, pp. 1338–1345, 2004, Doi: 10.1109/CEC.2004.1331052.
[10] S. LaValle, "Planning algorithms," *Cambridge university press*, 2006.
[11] J. M. Keil, "Decomposing a Polygon into Simpler Components," *SIAM Journal on Computing*, vol. 14, no. 4, pp. 799–817, Jul. 2006, Doi: 10.1137/0214056.
[12] C. Zhong *et al.*, "A Fast On-line Global Path Planning Algorithm Based on Regionalized Roadmap for Robot Navigation," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 319–324, Jul. 2017, Doi: 10.1016/J.IFACOL.2017.08.053.
[13] U. Orozco-Rosas, O. Montiel, and R. Sepúlveda, "Mobile robot path planning using membrane evolutionary artificial potential field," *Appl Soft Comput*, vol. 77, pp. 236–251, Apr. 2019, Doi: 10.1016/J.ASOC.2019.01.036.
[14] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "Heuristic approaches in robot path planning: A survey," *Rob Auton Syst*, vol. 86, pp. 13–28, Dec. 2016, Doi: 10.1016/J.ROBOT.2016.08.001.
[15] C. Lamini, S. Benhlima, and A. Elbekri, "Genetic Algorithm Based Approach for Autonomous Mobile Robot Path Planning," *Procedia Comput Sci*, vol. 127, pp. 180–189, Jan. 2018, Doi: 10.1016/J.PROCS.2018.01.113.
[16] A. J. Mohammed, K. I. Ghathwan, and Y. Yusof, "Optimal Robot Path Planning using Enhanced Particle Swarm Optimization algorithm," *Iraqi Journal of Science*, vol. 61, no. 1, pp. 178–184, Jan. 2020, Doi: 10.24996/IJS.2020.61.1.20.
[17] T. Younis Abdalla, A. Abed, A. A. Ahmed, T. Y. Abdalla, and A. A. Abed, "Path planning of mobile robot by using modified optimized potential field method," *Article in International Journal of Computer Applications*, vol. 113, no. 4, pp. 975–8887, 2015, Doi: 10.5120/19812-1614.
[18] T. F. Abaas and A. H. Shabeeb, "Autonomous Mobile Robot Navigation Based on PSO Algorithm with Inertia Weight Variants for Optimal Path Planning," in *IOP Conference Series: Materials Science and Engineering*, 2020, vol. 928, no. 2, p. 22005.
[19] R. Rashid, N. Perumal, I. Elamvazuthi, M. K. Tageldeen, M. K. A. A. Khan, and S. Parasuraman, "Mobile robot path planning using Ant Colony Optimization," *2016 2nd IEEE International Symposium on Robotics and Manufacturing Automation, ROMA 2016*, Feb. 2017, Doi: 10.1109/ROMA.2016.7847836.
[20] I. K. Ibraheem and F. H. Ajeil, "Path Planning of an autonomous Mobile Robot using Swarm Based Optimization Techniques," *Al-Khwarizmi Engineering Journal*, vol. 12, no. 4, pp. 12–25, Dec. 2016, Doi: 10.22153/KEJ.2016.08.002.
[21] G. Chen and J. Liu, "Mobile robot path planning using ant colony algorithm and improved potential field method," *Comput Intell Neurosci*, vol. 2019, pp. 1-10, 2019.
[22] W. Hou, Z. Xiong, C. Wang, and H. Chen, "Enhanced ant colony algorithm with communication mechanism for mobile robot path planning," *Rob Auton Syst*, vol. 148, p. 103949, 2022.
[23] J. Ou and M. Wang, "Path planning for omnidirectional wheeled mobile robot by improved ant colony optimization," *Chinese Control Conference*, *CCC*, vol. 2019-July, pp. 2668–2673, Jul. 2019, Doi: 10.23919/CHICC.2019.8866228.
[24] P. K. Das, S. K. Pradhan, S. N. Patro, and B. K. Balabantaray, "Artificial immune system based path planning of mobile robot," *in Soft Computing Techniques in Vision Science. Springer.*, vol. 395, pp. 195–207, 2012, Doi: 10.1007/978-3-642-25507-6_17/COVER.
[25] H. S. Dewang, P. K. Mohanty, and S. Kundu, "A Robust Path Planning For Mobile Robot Using Smart Particle Swarm Optimization," *Procedia Comput Sci*, vol. 133, pp. 290–297, Jan. 2018, Doi: 10.1016/J.PROCS.2018.07.036.

[26] M. A. Hossain and I. Ferdous, “Autonomous robot path planning in dynamic environment using a new optimization technique inspired by bacterial foraging technique,” *Rob Auton Syst*, vol. 64, pp. 137–141, Feb. 2015, Doi: 10.1016/J.ROBOT.2014.07.002.

[27] G. Wang, L. Guo, H. Duan, L. Liu, and H. Wang, “A bat algorithm with mutation for UCAV path planning,” *The Scientific World Journal*, 2012, Doi: 10.1100/2012/418946.

[28] W. Wang, M. Cao, S. Ma, C. Ren, X. Zhu, and H. Lu, “Multi-robot odor source search based on Cuckoo search algorithm in ventilated indoor environment,” *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, vol. 2016-September, pp. 1496–1501, Sep. 2016, Doi: 10.1109/WCICA.2016.7578817.

[29] Turki Y. Abdalla Ziyad T. Allawi, “An ABC-Optimized Reciprocal Velocity Obstacles Algorithm for Navigation of Multiple Mobile Robots,” *IRAQI JOURNAL OF COMPUTERS, COMMUNICATIONS, CONTROL AND SYSTEMS ENGINEERING*, vol. 15, no. 2, pp. 47-57, 2015.

[30] M. A. Kareem, T. A. Jaleel, and A. K. A. Hassan, “Collision Avoidance Using Cat Swarm Algorithm for Multi Mobile Robot Path Planning in Dynamic Environment,” *Iraqi Journal of Science*, vol. 57, no. 3C, pp. 2348–2359, 2016.

[31] F. Hason and S. Al-Darraji, “A review: On bio-inspired optimization methods for path planning of mobile robot,” *Iraqi Journal of Intelligent Computing and Informatics (IJICI)*, vol. 2, no. 1, pp. 1–10, 2023.

[32] T. K. Dao, T. S. Pan, and J. S. Pan, “A multi-objective optimal mobile robot path planning based on whale optimization algorithm,” *International Conference on Signal Processing Proceedings, ICSP*, pp. 337–342, Jul. 2016, Doi: 10.1109/ICSP.2016.7877851.

[33] X. Dai, S. Long, Z. Zhang, and D. Gong, “Mobile robot path planning based on ant colony algorithm with a∗ heuristic method,” *Front Neurorobot*, vol. 13, p. 15, Apr. 2019, Doi: 10.3389/FNBOT.2019.00015/BIBTEX.

[34] S. Hosseininejad and C. Dadkhah, “Mobile robot path planning in dynamic environment based on cuckoo optimization algorithm,” *Int J Adv Robot Syst*, vol. 16, no. 2, 2019.

[35] B. Khan, S. Nizamani, N. H. Kazi, S. Nizamani, and S. Nizamani, “A Prototype of Obstacle Avoidance for Autonomous Vehicle,” *Iraqi Journal of Science*, vol. 63, no. 5, pp. 2203–2210, May 2022, Doi: 10.24996/IJS.2022.63.5.33.

[36] J.-H. Zhang, Y. Zhang, and Y. Zhou, “Path planning of mobile robot based on hybrid multi-objective bare bones particle swarm optimization with differential evolution,” *IEEE Access*, vol. 6, pp. 44542–44555, 2018.

[37] X. Wang, Y. Shi, D. Ding, and X. Gu, “Double global optimum genetic algorithm–particle swarm optimization-based welding robot path planning,” *Engineering Optimization*, vol. 48, no. 2, pp. 299–316, 2016.

[38] F. Gul, W. Rahiman, S. S. N. Alhady, A. Ali, I. Mir, and A. Jalil, “Meta-heuristic approach for solving multi-objective path planning for autonomous guided robot using PSO–GWO optimization algorithm with evolutionary programming,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 7, pp. 7873–7890, Sep. 2020, Doi: 10.1007/S12652-020-02514-W.

[39] D. Połap and M. Woźniak, “Red fox optimization algorithm,” *Expert Syst Appl*, vol. 166, p. 114107, Mar. 2021, Doi: 10.1016/J.ESWA.2020.114107.

[40] M. Luneckas *et al.*, “Hexapod Robot Gait Switching for Energy Consumption and Cost of Transport Management Using Heuristic Algorithms,” *Applied Sciences*, vol. 11, no. 3, p. 1339, Feb. 2021, Doi: 10.3390/APP11031339.

[41] R. Natarajan, G. Megharaj, A. Marchewka, P. B. Divakarachari, and M. R. Hans, “Energy and Distance Based Multi-Objective Red Fox Optimization Algorithm in Wireless Sensor Network,” *Sensors*, vol. 22, no. 10, p. 3761, May 2022, Doi: 10.3390/S22103761.

[42] F. Hason and S. Al-Darraji, (in press), “Path Planning for an Autonomous Mobile Robot Using Red Fox Optimization Algorithm,” in *International Conference on Data Science and Intelligent Computing (ICDSIC’2022),* 2022.

[43] R. Pugal Priya, T. Saradadevi Sivarani, and A. Gnana Saravanan, “Deep long and short term memory based Red Fox optimization algorithm for diabetic retinopathy detection and classification," *Int J Numer Method Biomed Eng*, vol. 38, no. 3, p. e3560, Mar. 2022, Doi: 10.1002/CNM.3560.

[44] D. Połap and M. Woźniak, "Meta-heuristic as manager in federated learning approaches for image processing purposes," *Appl Soft Comput*, vol. 113, p. 107872, Dec. 2021, Doi: 10.1016/J.ASOC.2021.107872.

[45] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, Mar. 2014, Doi: 10.1016/J.ADVENGSOFT.2013.12.007.

[46] L. Doğan and U. Yüzgeç, "Robot path planning using gray wolf optimizer," in *Proceedings-International Conference on Advanced Technologies, Computer Engineering and Science (ICATCES'18)*, 2018.

[47] Lawrence J. Fogel, *Intelligence through simulated evolution: forty years of evolutionary programming*, 1999.

[48] M. A. Contreras-Cruz, V. Ayala-Ramirez, and U. H. Hernandez-Belmonte, "Mobile robot path planning using artificial bee colony and evolutionary programming," *Appl Soft Comput*, vol. 30, pp. 319–328, May 2015, Doi: 10.1016/J.ASOC.2015.01.067.