Bezier curve-based mobile robot path planning using the Proposed Enhanced Firefly Optimization algorithm

Prachi Bhanaria[1*], Praveen Kant Pandey[2] and Maneesha[2]
Email: [1]prachibhanariya30@gmail.com, [2]pkpandey.du@gmail.com, [2]maneesha@mac.du.ac.in
[1]Department of Electronic Science, University of Delhi, South Campus, India
[2]Maharaja Agrasen College, University of Delhi, India

Keywords: optimal path planning, firefly algorithm, static environment, mobile robot, Bezier curve.

Received: April 4, 2024

In the field of mobile robot navigation, the pursuit of optimal path planning and effective obstacle avoidance has evolved to unprecedented levels. This study introduces a novel hybrid methodology for mobile robot navigation, combining an enhanced Firefly Algorithm (FA) with Bezier curve-based path smoothing to achieve optimal trajectory planning and obstacle avoidance. The refined FA improves exploration dynamics by optimizing the fixed step size and attractiveness parameter, ensuring faster convergence and reduced computational overhead. Simultaneously, the Bezier curve technique facilitates smooth transitions, minimizing abrupt directional shifts and enhancing path continuity. The approach aims to determine the shortest and most efficient path between initial and target points, with control points in the Bezier curve playing a pivotal role in shaping trajectory fluidity and overall path length. Comparative evaluations against conventional FA methods and leading metaheuristic algorithms—such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO)—demonstrate the superior performance of the proposed framework. The method achieves a remarkable 3.98% reduction in average path cost over the Quadratic Polynomial technique and a 2.67% improvement over the Cubic Polynomial Equation-based approach, underscoring its effectiveness in delivering computationally efficient and dynamically optimized navigation solutions for mobile robots.

Povzetek: Študija predstavlja hibridno metodo za navigacijo mobilnih robotov, ki združuje izboljšani algoritem Firefly Optimization (FA) z glajenjem poti prek Bezierjevih krivulj. Metoda skrajša povprečne stroške poti v primerjavi s kvadratnimi polinomi in generira krajše ter bolj gladke poti z manjšim številom zavojev kot klasični FA in drugi metahevristični algoritmi.

1 Introduction

In nature, animals often work together in large groups, like birds flying in flocks, fish swimming in schools, or wolves hunting as a pack. These behaviours rely on teamwork, coordination, and self-organization to function smoothly. Similarly, Multi-Agent Systems (MAS) involve multiple simple agents working together to complete complex tasks, making systems more adaptable, reliable, and efficient. One of the biggest challenges in MAS is guiding multiple agents from their starting points to their destinations while following rules and avoiding obstacles. Effective coordination ensures each agent takes the best possible path, minimizing time and energy usage while optimizing performance for the entire system. In real-time applications, this requires fast and efficient path-planning algorithms to handle complex environments and quick decision-making [1][2].

Ayawli et al. and Gao et al. categorized path-planning methods into traditional search algorithms and intelligent algorithms. Traditional approaches, including A*, Artificial Potential Field (APF), and Rapidly-exploring Random Tree (RRT), focus on deterministic and heuristic-based searches [3]. In contrast, intelligent algorithms such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Grey Wolf Optimization (GWO), Marine Predators, Equilibrium Optimizers, Ant Colony Optimization (ACO), Firefly Algorithm (FA), etc., leverage adaptive and evolutionary strategies for optimization. Their study highlights the trade-offs between real-time performance and solution effectiveness across these methods [4].

Traditional path planning approaches, such as the A* algorithm, often struggle with efficiency and adaptability in dynamic settings. To address these limitations, Han Y. et.al. have proposed enhanced versions of A*, incorporating cost-based evaluation functions and safety distances to improve real-time decision-making. The integration of a Dynamic Window Approach (DWA) with stop-and-wait and replanning mechanisms has further enhanced responsiveness in multi-robot navigation, ensuring efficient and conflict-free transportation [5]. Genetic algorithms (GAs) have also been extensively utilized to optimize path planning. The Improved Genetic Algorithm (IGA) [6] leverages adaptive mutation and crossover operators to enhance search efficiency. Despite improvements, GAs still faces issues such as early convergence and path discontinuities, leading to suboptimal routes. The lack of a universally effective GA variation highlights the need for further refinement and hybrid approaches. For UAV path planning, a quantitative framework for evaluating LGPSO's (Local-Global

Particle Swarm Optimization) performance has been developed by Cheng et.al., incorporating metrics such as smoothness and efficiency scores. LGPSO demonstrates effectiveness in optimizing UAV paths but is constrained by limited real-world environmental modelling, necessitating improvements for practical applications [7]. NI-GWO (Nonlinear Improved Grey Wolf Optimization) algorithm, integrated with DWA, enhances UAV trajectory optimization and dynamic obstacle avoidance. However, its reliance on simulated data indicates a need for real-world validation to ensure robustness in diverse environments [8]. Marine Predator Algorithm (MPA) modifications have led to the Multi-Objective Marine Predator Algorithm (MMPA), which improves UAV path optimization over classical techniques like PSO, DE, and GWO. However, its single-objective approach oversimplifies real-world UAV path planning, necessitating a shift towards multi-objective formulations for greater adaptability [9]. The Equilibrium Optimizer (EO) introduces physics-inspired mass equilibrium principles for path planning, yet it suffers from population diversity issues and a tendency toward local optima. The introduction of an improved version, SCEO (Self-Adaptive EO) [10], enhances the balance between exploration and exploitation, showing promise in mobile robot path planning but requiring further validation in complex engineering applications. Ant Colony Optimization (ACO) [11] has been improved through an Intelligent Enhanced ACO (IEACO), which incorporates pheromone distribution optimization, adaptive state transition strategies, and multi-objective evaluation frameworks. Despite its strengths, IEACO still requires integration with curve optimization techniques such as B-spline curves to ensure smooth path trajectories for mobile robots.

The Firefly Algorithm (FA) offers a robust alternative to the approaches mentioned above, leveraging swarm intelligence principles inspired by fireflies' bioluminescent attraction behaviour. Unlike GA, FA maintains a better balance between exploration and exploitation, reducing the likelihood of premature convergence. Compared to PSO-based methods [12] like LGPSO, FA exhibits superior adaptability to dynamic environments, making it ideal for real-time UAV and multi-robot path planning. Furthermore, FA avoids the stagnation issues observed in EO and NI-GWO, ensuring a more diverse population and enhanced global search capabilities. When applied to mobile robot navigation, FA outperforms A* and ACO variants by dynamically adjusting step sizes and attraction mechanisms, leading to smoother and more efficient paths. Its ability to integrate adaptive hybridization techniques makes it highly suitable for real-world applications requiring precise, conflict-free, and computationally efficient path planning.

Wahab et al. [13] enhance exploration within the search space, reinforcing its swarm-based meta-heuristic capabilities. Reducing the parameter 'α' refines potential solutions by decreasing the step size, thereby strengthening the exploitation phase. This approach maintains a balance between exploring the search space and effectively refining promising solutions. During the algorithm's execution, the parameters 'β' and 'γ' remain unchanged. By combining the rapid convergence of the Firefly algorithm with the straightforward nature of Quadratic parametric equations, the proposed method efficiently determines the shortest and smoothest trajectory for the mobile robot [14]. Sura Mazin Ali et al. [15] introduced a technique for identifying collision-free points by leveraging the Firefly algorithm. This approach utilizes cubic polynomial trajectory equations to generate smooth paths within a specified timeframe. When tested in various simulated environments with different levels of complexity, the proposed method effectively navigated obstacles, optimized collision-free paths, and enabled the mobile robot to seamlessly follow the planned path.

Our Proposed Algorithm: To enhance the Firefly Algorithm (FA), we apply adaptive parameter adjustments to balance exploration and exploitation effectively. The light absorption coefficient (γ) is adapted dynamically to support broad exploration in early stages and refined exploitation later. Additionally, the attractiveness parameter (β_0) is also modified to 0.5 instead of being 1 in the standard firefly algorithm. Initially set high to encourage diverse exploration, β_0 decreases to focus on exploitation, preventing premature convergence. Furthermore, closer fireflies experience stronger attraction forces, while distant ones have reduced attraction, maintaining a balance between exploration and exploitation. Similarly, the step size (δ) is decreased to optimize FA performance. Fireflies with higher fitness values take smaller steps for fine-tuning solutions, while those with lower fitness values take larger steps to explore the search space more effectively. This dynamic adaptation of parameters ensures that the FA maintains diversity, accelerates convergence, and improves solution quality.

The layout of the paper is introduced here. The proposed FA with standard Firefly algorithm features is described in Section 2. The Bezier curve model to generate a smooth path after the FA path is mentioned in section 3. Section 4 provides the simulation results obtained by applying Bezier curve-based FA to minimize the Path cost in suggested environments. The conclusion and future trends are given in Section 5.

2 Firefly algorithm

The Firefly Algorithm (FA) is a meta-heuristic technique inspired by swarm intelligence. It operates based on the social interaction and communication behaviour of fireflies, particularly their luminescent flashing patterns. This algorithm is built upon a simplified model of how fireflies behave and incorporates three fundamental principles:

1. Fireflies are unisex, meaning they are attracted to one another regardless of gender.
2. The attraction is determined by brightness; a dimmer firefly moves toward a brighter one, while random movement occurs when two fireflies have equal brightness.

3. The firefly's brightness correlates with the objective function's value.

The FA simulates the social behaviour of fireflies by using flashing lights primarily to attract mates or deter predators. It relies on the principle that light intensity, denoted as I, decreases with increasing distance r. Additionally, the surrounding environment's absorption of light increases as the distance from the source grows. In this algorithm, light intensity is tied to the fitness function of the optimization problem being solved.

To define the firefly's attractiveness, we use the following parameters:

- β_0 represents the attractiveness of a firefly when r = 0.
- Γ is the coefficient governing the absorption of light.
- x_i and x_j are the positions of fireflies I and j, respectively.
- r_{ij} is the distance between the two fireflies.
- d is the dimension of the search space.
- t represents the iteration count.
- α is the randomization parameter.

The attractiveness of a firefly is expressed by the equation 1:

$$\beta = \beta_0 e^{-\gamma r^2} \quad (1)$$

Thus, it will vary with the Euclidean distance r_{ij} between firefly i and firefly j as defined in equation 2:

$$r_{ij} = \|x_i - x_j\| = \sum_{k=1}^{d} (x_{i,k} - x_{j,k}) \quad (2)$$

Where d is the spatial dimension, x_i, k is the k-th component of firefly i in the d-dimensional space. For any given two fireflies xi and x_j, the movement of the firefly i is attracted to another firefly j is determined using equation 3.

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r^2}(x_j - x_i) + \alpha_t * \delta_t \quad (3)$$

This equation governs the position updates of fireflies based on attraction, randomness, and optimization within the problem space.

Modified firefly algorithm:

In the Standard firefly algorithm, the maximum attraction parameter is typically set to 1. However, an analysis of the movement formula (Equation 3) reveals that such a high attraction value causes fireflies to rapidly converge toward the current best individual in just a few iterations. While this accelerates optimization, it also leads to premature loss of population diversity.

To mitigate this issue, the maximum attraction parameter can be reduced to 0.5. This adjustment ensures that Firefly's approach to brighter individuals is more gradual, preserving population diversity throughout the iterations. As a result, the attraction strength no longer converges to 1 but instead stabilizes at 0.5, allowing sustained exploration and a higher likelihood of locating the global optimum. The updated position equation is defined by Equation 4.

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r^2}(x_j - x_i) + \alpha_t * \delta_t * (rand - 0.5) * (UL - LL) \quad (4)$$

While moving, the robot will continuously check for obstacles using the following equation 5:

$$DistRO = \sqrt{(O_1 - R_1)^2 + (O_2 - R_2)^2} \quad (5)$$

Where DistRO is the distance between the robot and an obstacle, (O_1, O_2) are the coordinates of an obstacle and (R_1, R_2) are the coordinates of the robot.

The light absorption coefficient (γ) is used to approximate light absorption for the solution of each standard firefly method. The firefly's attractiveness is directly proportional to the value of the relevant fitness function. Use of **adaptive γ** that allows wider exploration in the early stages and focuses on exploitation in later stages.

Attractiveness parameters: The attractiveness parameter (β_0) controls how strongly a firefly is drawn toward another. Choosing an appropriate β_0 is crucial for maintaining diversity and avoiding premature convergence. To enhance FA, using β_0=0.5, which starts high to encourage exploration and gradually decreases to favour exploitation.

Step size (δ): In FA, the efficiency of the solution depends on the step size parameter. The step size determines how far a firefly moves during each iteration. A static step size may hinder convergence speed or exploration ability. To enhance FA, using step size in the position update equation, where fireflies with better fitness values take smaller steps for fine-tuning, while those with poorer fitness take larger steps to explore more broadly. Table 1 shows the parameters used for the Modified Firefly Algorithm, and the Flowchart of the proposed Firefly algorithm is shown in Figure 1.

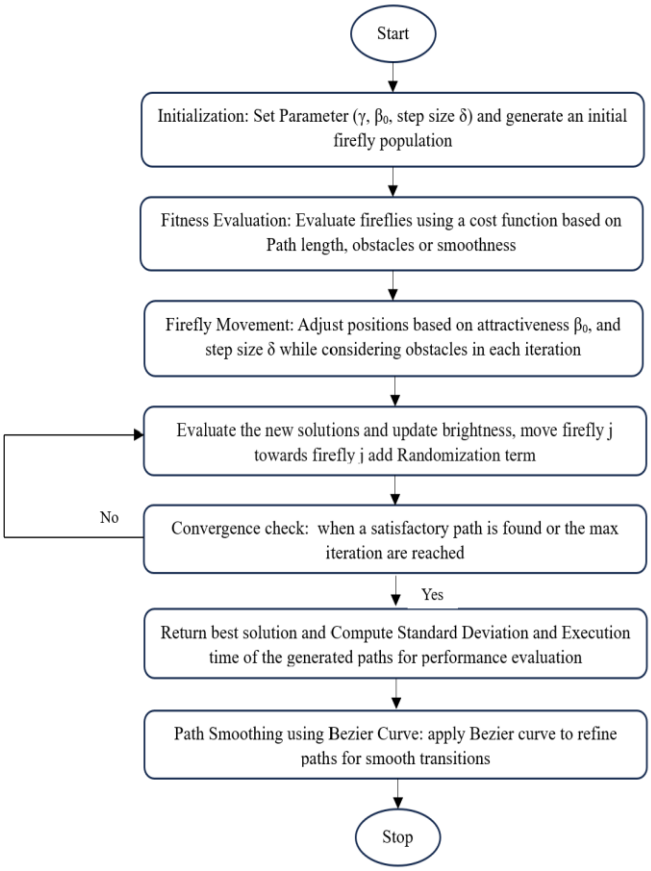


Figure 1: Shows the flowchart of the proposed Firefly Algorithm.

3 Bezier Curve-based smooth path

Bezier curves are widely used in computer graphics for drawing shapes, animating elements in CSS, and various other applications. One key benefit of Bezier curves is their convexity—if the control points form a convex polygon, the curve itself remains convex. These curves are mathematically defined based on control point coordinates P = (x, y) and are calculated using an equation that depends on a parameter t, which ranges from 0 to 1, determining the curve's shape.

Their widespread adoption means that designers and developers can rely on consistent behavior and tools for manipulating Bezier curves across different software applications. Bezier curve is specifically designed to produce smooth curves between control points. The interpolation algorithm used in Bezier curves ensures that the resulting curve is continuous and smooth, even when control points are moved. On the other hand, while cubic polynomial equations can also generate smooth curves, achieving this smoothness may require additional effort and careful selection of coefficients. A Bezier curve of degree n is a parametric curve of basic polynomials of degree n, and it can be defined as:

$$\sum_{i=0}^{n} P_i B_{i,n}(t) \qquad t \in [0,1] \qquad (4)$$

Table1: Modified FA parameters.

Parameters	Modified FA
Iteration/Generations	50
No. of fireflies	20
Randomization parameter (α)	0.625
Attractiveness (β)	0.5
Step size (δ)	0.125
Light absorption coefficient (γ)	0.99

Where $B_{i,n}(t)$ is a Bernstein polynomial. This paper uses the Bezier curve at every intersection point of two straight segments of the path to gradually change the direction, and maintain the longest possible line between these curves, as shown in Figure 2.

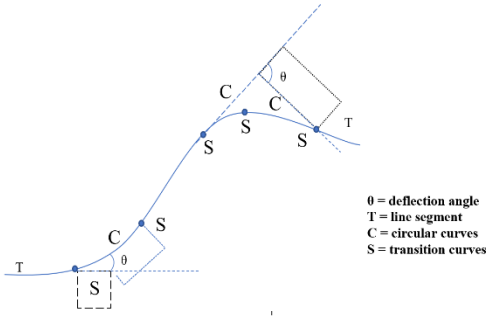


Figure 2: Smooth path using Bezier curve

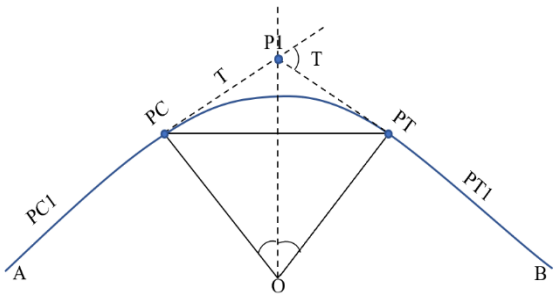


Figure 3: Choosing the control points

Figure 3 shows the smooth control points where PCI P1 = PT1 P1, PCP1 = PT P1, and there is a collision-free path between PC and PT. Additionally, Bezier curves allow **local control**, meaning adjustments to one control point only affect a portion of the curve—an essential feature for real-time path corrections.

In contrast, **quadratic polynomials** generate parabolic trajectories defined by second-degree equations. While computationally efficient, they lack the flexibility needed for complex paths, as they can only produce simple curves with a single inflection point. **Cubic polynomials**, offering third-degree equations, provide better flexibility with two inflection points, making them suitable for smoother transitions in robotic motion planning. However, even cubic polynomials struggle with highly

dynamic paths, as they may produce unintended oscillations if not properly constrained.

When comparing these methods, Bezier curves outperform polynomials in several key areas. First, they offer infinite smoothness (C^{∞} *continuity*), whereas quadratic and cubic polynomials are limited to C^1 and C^2 continuity, respectively. This makes Bezier curves ideal for applications requiring jerk-free motion, such as autonomous vehicles and precision robotics. Second, Bezier curves can model complex, multi-turn paths by simply adjusting control points, whereas polynomials require higher degrees (and thus more computational effort) to achieve similar complexity. While polynomials are computationally cheaper, Bezier curves balance path quality and adaptability better, especially in environments requiring dynamic adjustments, and the comparative parameters are shown in Table 2.

Table 2: Features of bezier curve and quadratic/ cubic polynomial

Features	**Bezier Curves**	**Quadratic/Cubic Polynomial**
Design Control	Local & intuitive	Global change affects all
Smooth Curve Joining	Easy (splines)	Harder (C^2 continuity)
Numerical Stability	Stable	Prone to oscillations
Parametric Form	Yes	Often scalar (y = f(x))
Efficiency (for simple cases)	Needs more computation	Often faster

4 Simulation results

In this study, various advanced optimization algorithms have been explored, including Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Grey Wolf Optimizer (GWO), Marine Predators Algorithm (MPA), Equilibrium Optimizer (EO), Ant Colony Optimization, etc. These algorithms have demonstrated significant performance in solving optimization problems across different domains. However, they also exhibit certain limitations. To validate the effectiveness of the proposed Firefly Algorithm (FA), simulation experiments were conducted, comparing its performance with pre-existing FA, Variants of Ant Colony Optimization, PSO, and Genetic Algorithm. The results indicate that the Proposed FA achieves superior Path length, computational time, and Number of turns.

To check the optimality of the proposed FA in terms of path cost for navigation, the different suggested environments have been simulated using Python. To develop the proposed Firefly Algorithm, we select Firefly parameters according to the problem domain. The randomization parameter (α), attractiveness parameter (β), and the Light absorption coefficient (γ) are considered between the ranges of 0 to 1. These control parameters are finalized as $\beta = 0.5$, $\gamma = 0.99$, and $\alpha = 0.625$, forming an optimization after 50 times (iterations, and the number of fireflies (Population size) as 20.

This study evaluates the proposed method in two-dimensional mobile robot environments, represented by (x, y) coordinates, which include static obstacles. The Bezier curve-based Firefly optimization algorithm is implemented across five different robot environments or maps to determine a collision-free path for the mobile robot.

The performance of the proposed Bezier curve-based Firefly optimization algorithm is compared with the Cubic polynomial-based Firefly algorithm, as proposed by **Noor Alhuda F. Abbas [9] and Sura Mazin Ali et al. in 2022 [10].** Table 3 compares the two algorithms' results for the five maps and shows that the proposed Bezier curve-based Firefly optimization algorithm gives better results and produces smoother paths.

The table presents a comparative analysis of different FA Optimization methods: Quadratic Polynomial-based, Cubic Polynomial Equation-based, and Proposed Firefly Bezier Curve-based, applied across five different mapping scenarios. The cost function values illustrate the effectiveness of each method, with the Bezier Curve-based Proposed FA Optimization method consistently achieving lower costs compared to the other two approaches.

For instance, in Map 1, the Bezier Curve-based method resulted in a cost function value of **25.214**, representing an **improvement of 1.01%** over the Quadratic Polynomial-based method and **0.89%** over the Cubic Polynomial Equation-based method. The most significant enhancement appears in Map 5, where the Bezier Curve-based approach yielded a cost function of **18.333**, marking an **11.74%** reduction compared to the Quadratic Polynomial method and **7.64%** compared to the Cubic Polynomial Equation-based method.

Thus, the Bezier Curve-based FA Optimization method demonstrated an average cost reduction of approximately **3.98%** compared to the Quadratic Polynomial-based approach and **2.67%** compared to the Cubic Polynomial Equation-based approach, signifying its superior efficiency in minimizing the cost function values across different mapping scenarios, as shown in Table 4.

Table 3: Comparison between the FA Optimization method (proposed by Noor Alhuda F. Abbas [14]) and the FA Optimization method (proposed by Sura Mazin Ali et al. [15]), and the Proposed FA based on Bezier curve

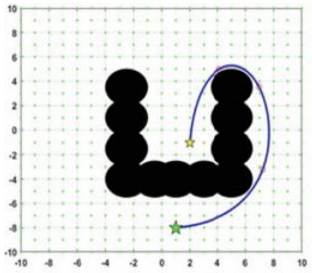
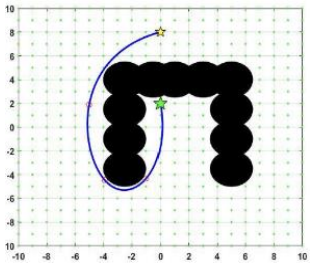
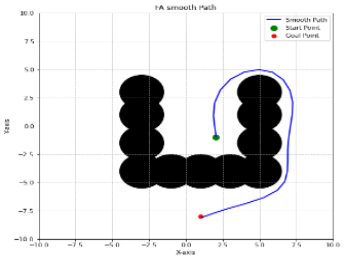
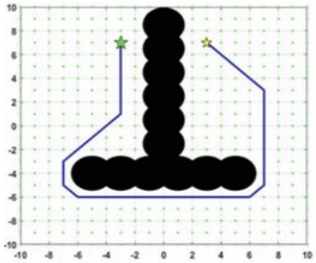
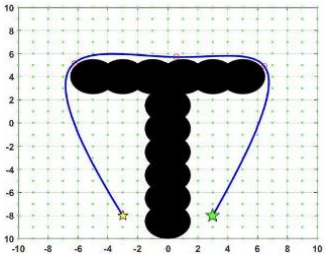
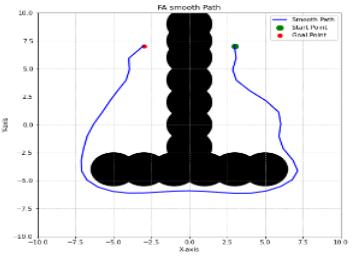
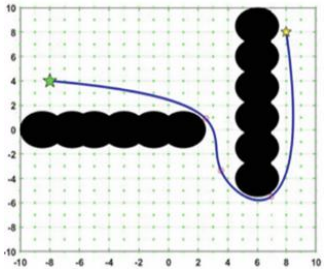
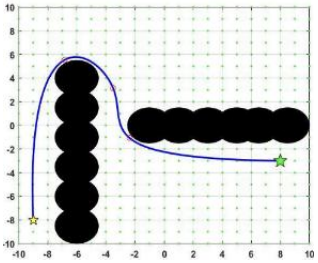
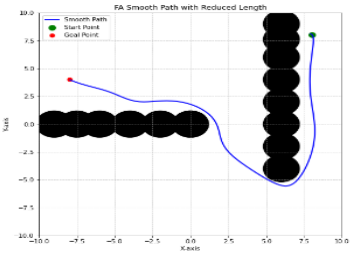
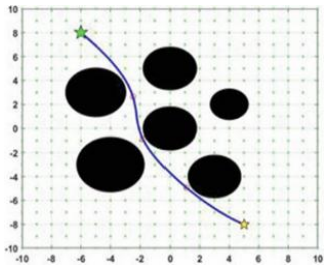
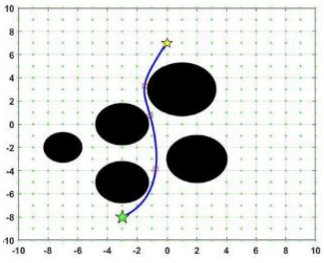
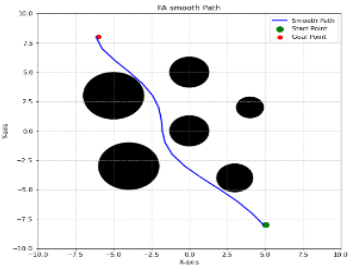
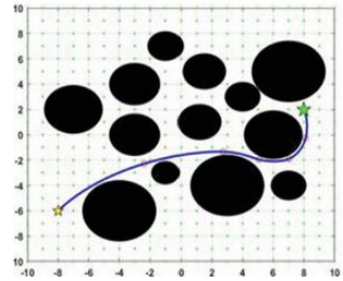
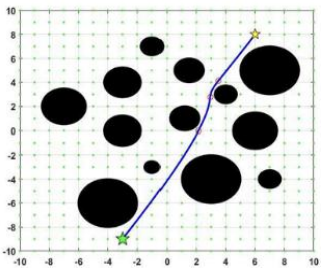
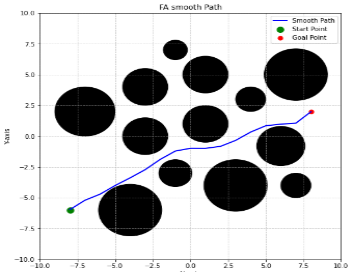
Map	Quadratic polynomial-based FA [9]	Cubic Polynomial Equation-based FA [10]	Bezier Curve-based Proposed FA
1			
2			
3			
4			
5			

Table 4: Simulation results by Noor Alhuda F. Abbas [14]) and Sura Mazin Ali et al. [15]) and proposed FA based

Map	Start	End	Cost Function using Quadratic polynomial-based FA Optimization method (proposed by Noor Alhuda F. Abbas [14])	Cost Function using Cubic Polynomial Equation based FA Optimization method (proposed by Sura Mazin Ali et al. [15])	Cost Function using Bezier Curve-based FA Optimization method	Improvement in Cost Function compared to [14]	Improvement in Cost Function compared to [15]
1	(2, -1)	(1, -8)	25.473	25.442	25.214	**1.01%**	**0.89%**
2	(3, 7)	(-3, 7)	38.887	37.953	37.643	**3.19%**	**0.81%**
3	(8, 8)	(-8, 4)	34.05	34.186	32.809	**3.64%**	**4.027%**
4	(5, -8)	(-6, 8)	20.071	20.0062	20.0047	**0.330%**	**0.0074%**
5	(-8, -6)	(8,2)	20.773	19.8418	18.333	**11.74%**	**7.64%**

This research also includes a thorough comparative analysis between the proposed Firefly algorithm and several well-established path planning techniques. The main goal was to evaluate the performance of these algorithms, particularly in relation to PFA, which is designed as a more advanced and optimized approach for complex path planning tasks. The assessment covers algorithms such as Ant Colony Optimization (ACO), Genetic Algorithm (GA), A*, and Rapidly Exploring Random Tree (RRT). To ensure accurate evaluation, average values and standard deviations of key performance metrics, including path length and execution time, were analyzed.

Table 5 presents a comparison of multiple paths planning algorithms, including Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Genetic Algorithm (GA), A*, Rapidly-exploring Random Tree (RRT), and Path-Finding Algorithm (PFA). The analysis evaluates key performance metrics such as path length, execution time, and the number of turns to assess their effectiveness.

Among the algorithms evaluated, the PFA method demonstrates the most efficient performance in terms of path length and execution time. With an average path length of **117.63 m**, it outperforms all other approaches, achieving the shortest path while maintaining a minimal deviation. Additionally, PFA exhibits the lowest execution time, taking only **7.0108 sec**, which is significantly faster compared to ACO (**32.12 sec**) and PSO (**18.09 sec**). The number of turns in PFA is remarkably low at **2**, highlighting its capability to generate smooth and optimized paths.

On the other hand, ACO results in the longest path length (**159.51 m**) and the highest number of turns (**44**), suggesting a more complex and indirect route compared to the other algorithms. While GA achieves a relatively shorter path length (**130.09 m**) than PSO (**132.84 m**), its execution time is slightly lower at **16.89 sec**, making it a more time-efficient alternative. Meanwhile, A* and RRT maintain moderate path lengths (**134.98 meters** and **135.81 m**, respectively), with A* offering faster execution (**8.95 sec**) compared to RRT (**12.21 sec**).

Overall, PFA emerges as the most effective algorithm in terms of achieving the shortest and smoothest path while minimizing execution time, making it a promising choice for path planning optimization.

Table 5: Simulation results for different path planning algorithms [12] and Proposed FA

Parameter	PSO	ACO	GA	A*	RRT	PFA
Path length (m)	132.84±0.78	159.51±3.65	130.09±1.49	134.98±1.67	135.81±1.21	**117.63±0.0059**
Execution Time(s)	18.09±2.43	32.12±1.44	16.89±0.98	8.95±0.34	12.21±2.31	**7.0108 ±1.72**
Number of turns	13	44	17	9	222	**2**

5 Conclusion

The present work introduced a novel Bezier Curve-based Firefly Algorithm optimization method for efficient path planning in Multi-Agent Systems with improved path optimization. Further, the proposed algorithm enhances the balance between exploration and exploitation by dynamically adapting various parameters of the standard Firefly Algorithm.

The proposed model is rigorously evaluated through comprehensive simulation experiments, which involve multiple map scenarios to demonstrate remarkable improvements in path planning performance metrics. The results confirm that the Bezier Curve-based Firefly Algorithm achieved substantial reductions in the cost function compared to existing methods. In the most complex scenario (i.e. Map 5), the proposed algorithm reduced path cost to 18.333, marking an 11.74% improvement over the Quadratic Polynomial-based method and a 7.64% improvement compared to the Cubic Polynomial-based method. Averaging across multiple scenarios, the proposed Bezier Curve-based FA delivered approximately 3.98% better path costs than Quadratic Polynomial methods and about 2.67% better than Cubic Polynomial-based methods. Comparative analyses with other established path-planning algorithms such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Genetic Algorithm (GA), A*, and Rapidly-exploring Random Tree (RRT) further validated the efficiency of the proposed model. The proposed model demonstrated an optimal path length averaging 117.63 meters, notably shorter than 130.09 meters (GA), 159.51 meters (ACO), and 132.84 meters (PSO). Additionally, the proposed method achieved the fastest execution time, averaging approximately 7.01 seconds as compared to ACO's 32.12 seconds and PSO's 18.09 seconds. Furthermore, the proposed model significantly improved path smoothness, achieving an average of only 2 turns per path as compared to other algorithms, such as ACO with 44 turns and RRT with 222 turns. These improvements are attributed to the inherent flexibility and local control provided by the Bezier curve formulation, ensuring smooth trajectories vital for real-time robotic applications

Overall, the proposed Bezier Curve-based FA method shows an innovative and highly effective solution for real-time, collision-free path planning model, which particularly suitable for applications that demand precise manoeuvring, rapid computation, and robust obstacle avoidance capabilities.

Future research could focus on extending this methodology to dynamic three-dimensional environments and validating its effectiveness through real-world experiments

References

[1] Abujabal, N., Fareh, R., Sinan, S., Baziyad, M., Bettayeb, M. A Comprehensive Review of the Latest Path Planning Developments for Multi-Robot Formation Systems. Robotica 2023, 41, 2079-2104. https://doi.org/10.1017/s0263574723000322

[2] Lin, S., Liu, A., Wang, J., Kong, X, A Review of Path-Planning Approaches for Multiple Mobile Robots. Machines, 2022, 10, 773. https://doi.org/10.3390/machines10090773

[3] Kiadi, M., García, E., Villar, J. R., & Tan, Q., A*-Based Co-Evolutionary Approach for Multi-Robot Path Planning with Collision Avoidance. Cybernetics and Systems, 2022, 54(3), 339–354. https://doi.org/10.1080/01969722.2022.203000

[4] Shami, T.M., El-Saleh, A.A., Alswaitti, M., Al-Tashi, Q., Summakieh, M.A., Mirjalili, S, Particle Swarm Optimisation, A Comprehensive Survey. IEEE Access 2022, 10, 10031–10061. https://doi.org/10.1109/access.2022.3142859

[5] Han, Y.; Li, C.; An, Z. Based on the Integration of the Improved A* Algorithm with the Dynamic Window Approach for Multi-Robot Path Planning. Appl. Sci. 2025, 15, 406. https://doi.org/10.3390/app15010406

[6] Zhang, Z.; Yang, H.; Bai, X.; Zhang, S.; Xu, C. The Path Planning of Mobile Robots Based on an Improved Genetic Algorithm. Appl. Sci. 2025, 15, 3700. https://doi.org/10.3390/app15073700

[7] Cheng, Qing & Zhang, Zhengyuan & Du, Yunfei & Li, Yandong. Research on Particle Swarm Optimization-Based UAV Path Planning Technology in Urban Airspace. Drones, 2024, 8. 701. 10.3390/drones8120701.

[8] Zhou, X.; Shi, G.; Zhang, J. Improved Grey Wolf Algorithm: A Method for UAV Path Planning. Drones 2024, 8, 675. https://doi.org/10.3390/drones8110675

[9] Gong, Rong & Gong, Huaming & Hong, Lila & Li, Tanghui & Xiang, Changcheng, A novel marine predator algorithm for path planning of UAVs. The Journal of Supercomputing. 81, 2025. 10.1007/s11227-025-07002-6.

[10] Liu, Yuting & Ding, Hongwei & Wang, Zongshan & Dhiman, Gaurav & Yang, Zhijun & Hu, Peng, An Enhanced Equilibrium Optimizer for Solving Optimization Tasks. Computers, Materials & Continua, 77, 2023, 2385-2406. 10.32604/cmc.2023.039883.

[11] Li, P.; Wei, L.; Wu, D. An Intelligently Enhanced Ant Colony Optimization Algorithm for Global Path Planning of Mobile Robots in Engineering Applications. Sensors, 2025, 25, 1326. https://doi.org/10.3390/s25051326

[12] Poy, Y.-L., Loke Z.-Y., Darmaraju, S., Goh, C.-H., Kwan, B.-H., Liu, H., Ng, D.W.K, Enhanced Particle Swarm Optimisation for Multi-Robot Path Planning with Bezier Curve Smoothing. Robotics 2024, 13, 141. https://doi.org/10.3390/robotics13100141

[13] Ab Wahab MN, Nazir A, Khalil A, Bhatt B, Mohd Noor MH, et al., Optimised path planning using Enhanced Firefly Algorithm for a mobile robot, 2024 PLOS ONE 19(8): e0308264. https://doi.org/10.1371/journal.pone.0308264

[14] Abbas, N.A.F, "Mobile Robot Path Planning Optimization Based on Integration of Firefly Algorithm and Quadratic Polynomial Equation" In: Tran DT., Jeon G., Nguyen T.D.L., Lu J., Xuan TD. (eds)., Intelligent Systems and Networks, ICISN 2021. Lecture Notes in Networks and Systems, 243, Springer, Singapore. DOI 10.1007/978-981-16-2094-2_64, 2021.

[15] Sura Mazin Ali, Janan Farag Yonan, Omar Alniemi, Amjed Abbas Ahmed, "Mobile Robot Path Planning Optimization Based on Integration of Firefly Algorithm and Cubic Polynomial Equation", J. ICT Res. Appl., Vol. 16, No. 1, 2022, 1-22. https://doi.org/10.5614/itbj.ict.res.appl.2022.16.1.1