European Journal of Research Development and Sustainability (EJRDS)
Available Online at: https://www.scholarzest.com
Vol. 7 No 03 March 2026
ISSN: 2660-5570

ENHANCED MOTION PLANNING FOR MOBILE ROBOTS USING A HYBRID A*-FPA FRAMEWORK

Ibtehal .M. Habeeb, Alia K.abdulhasan
Computer Science Department, University of Technology, Baghdad, Iraq
**cs.20.32@grad.uotechnology.edu.iq* , hassanalia2000@yahoo.com

Article history:

Received: 11th February 2026
Accepted: 10th March 2026

Abstract:

Motion planning is one of the most basic functions for robots, which helps robots move safely and securely through cluttered and changing spaces. While A* remains the most popular approach for generating optimal paths, its performance is affected for cluttered and changing spaces. To overcome these limitations, the current research suggests the hybrid approach by combining A* and FPA. The hybrid approach is intended to improve the quality of paths by smoothing the paths, reducing unnecessary turns, and minimizing computational costs. The results obtained through experimentation prove that the hybrid approach of A* and FPA is more effective and results in shorter and more coherent paths than A* search algorithms.

Keywords: A* Algorithm, Flower Pollination Algorithm, Hybrid Optimization, Motion Planning and Robotics

1. INTRODUCTION

Motion planning is a fundamental problem in artificial intelligence and robotics that seeks to find a safe and efficient route for a robot to travel from a given state to a desired state while avoiding obstacles (Lynch & Park, 2019). A* is considered a leading search algorithm in this area due to its balance of optimality and computational efficiency. Nevertheless, it is known to perform less satisfactorily in large and/or irregular environments. As a result, there is a growing trend of combining various bio-inspired optimization techniques, such as FPA, with A*, in order to develop a more adaptive and robust motion planner (Yang, 2012). Such a hybrid search paradigm is believed to hold significant promise in creating a motion planner that is capable of dealing with uncertainty and complexity in a practical environment.

2. RELATED WORK

Several research works have focused on improving classical path planning methods such as Dijkstra, A*, and RRT, which form the basis of current-day robot navigation systems (LaValle, 2006; Choset et al., 2005; Lynch & Park, 2017). In response to the drawbacks of local minima and excessive computational complexity, nature-inspired metaheuristic methods have gained considerable research focus. Among these methods, the Flower Pollination Algorithm (FPA), developed by Yang (2012), has been recognized as an efficient method of global optimization inspired by nature through pollination of plants. Further enhancements of this method have been made to improve its global optimization capability and convergence properties (Cui, Sun & Wang, 2018; Sahar & Hasan, 2020), and its effectiveness has been verified through various surveys on optimization problems, including path planning and decision problems (Abdel-Basset et al., 2018).

Research works on employing FPA for robot navigation have proven its capability to generate smooth and efficient paths through complex environments, even outperforming classical heuristics on occasions (Li & Zhang, 2016; Mohan & Al-Khazraji, 2019). At the same time, hybrid optimization frameworks employing heuristic methods and metaheuristic methods such as A*, GA, and PSO have been recognized to provide considerable improvements to exploration efficiency and optimality of paths generated through complex environments (Zhang et al., 2015; Xia & Zhou, 2017; Riaz et al., 2020). Recent research works on hybrid optimization methods have recognized its growing role in robot navigation and emphasized its significance through hybrid optimization models of path planning problems (Patel & Xiang, 2022; Alam et al., 2023).

Nevertheless, very few research works have focused on employing hybrid optimization methods such as A* and FPA. As mentioned by Sahu and Padhy (2021), this hybrid method has considerable potential to improve robot motion planning performance through its ability to balance exploration with local refinement and is a promising research direction to improve the performance of hybrid optimization methods.

3. MOTION PLANNING

Motion planning is defined as "the computation of a feasible path that guides the robot from its initial state to its desired goal state while satisfying constraints such as obstacles and physical constraints like joint limits and torque limits."

There are various ways of defining motion planning. These include path planning and motion planning; online and offline methods; and optimal and satisficing methods. When the environment is represented as a graph, it is possible to use A* algorithm to compute a cost-effective path with the help of heuristic evaluation.

4.GRAPH SEARCH

The use of free space graphing makes it possible to utilize formal search strategies to find a viable path between the initial state and the goal state. Among the graph-based search strategies, A* search is unique in its ability to find an optimal balance between cost minimization and heuristic search, thus encapsulating some of the most prominent path planning strategies for robots.

4.1 The A* Algorithm

Here, A* behaves as a best-first search algorithm, which assigns scores to nodes based on the f(n) function, where f(n) = g(n) + h(n), and g(n) represents the total cost from the initial node to the current node, while h(n) represents an estimate of the remaining cost to reach the goal node (Lynch & Park, 2019). The OPEN list is accessed repeatedly to choose nodes based on the minimum f(n). Once this node is expanded, neighboring nodes are assessed based on their tentative costs, and they are inserted into the OPEN list according to their scores. This cycle continues until the goal node is reached. Although it is impossible to calculate an accurate h(n), this heuristic can significantly improve the efficiency of this search algorithm. Although A* is an optimal path search algorithm, it is often observed that the resulting path is not smooth and is quite irregular in nature, which is why optimization algorithms such as the Firefly Path Algorithm (FPA) are often integrated into this search algorithm (Yang, 2012).

The algorithm works with this data structure, consisting of OPEN, the cost matrix, the array of past_cost values, and the search tree. The first node of OPEN is removed and designated as the current node, and its cost is calculated using the optimistic heuristic function. If the current node belongs to the goal set, then the algorithm terminates, and the path is constructed following the parent pointers. Otherwise, the tentative_past_cost of each neighboring node is made consistent, and the neighboring node is put back into the OPEN list according to its estimated total cost. Then, the process returns to the beginning of the main loop, removing the first node of OPEN and assigning it to the current node. If the heuristic function for calculating the cost-to-go is exact, then the algorithm will use the minimum number of nodes necessary to solve the problem.

Nevertheless, it is not feasible to compute the exact heuristic cost-to-go, so the heuristic cost-to-go must be computed efficiently and as close to the actual cost-to-go as possible to obtain optimal results. This method can guarantee an optimal path if the heuristic function is admissible. Though A* is computationally efficient, its results for grid-based maps can yield non-smooth paths, which are not suitable for actual robot movement. To overcome these disadvantages, an optimization method such as the FPA can be employed to refine the path (Yang, 2012).

Algorithm 4.1 A* Search

```
1: OPEN ← {1}
2: past_cost[1] ← 0, past_cost[node] ← infinity for node ∈ {2, ..., N}
3: while OPEN is not empty do
4:   current ← first node in OPEN, remove from OPEN
5:   add current to CLOSED
6:   if current is in the goal set then
7:     return SUCCESS and the path to current
8:   end if
9:   for each nbr of current not in CLOSED do
10:    tentative_past_cost ← past_cost[current]+cost[current,nbr]
11:    if tentative_past_cost < past_cost[nbr] then
12:      past_cost[nbr] ← tentative_past_cost
13:      parent[nbr] ← current
14:      put (or move) nbr in sorted list OPEN according to
             est_total_cost[nbr] ← past_cost[nbr] +
                 heuristic_cost_to_go(nbr)
15:    end if
16:  end for
17: end while
18: return FAILURE
```

4.2 Grid Methods

Discretizing the configuration space is a fundamental step when applying search-based planners like A*. One of the simplest approaches is to impose a grid structure, where each dimension of the configuration space is sampled uniformly. If an n-dimensional space is discretized into k points per dimension, the resulting grid contains k^n nodes. An A* planner operating on a grid is considered resolution-complete, meaning it will find a valid solution whenever one exists at the chosen resolution. The generated path corresponds to the shortest route consistent with the allowable movements on the grid.

An A* grid-based path planner is resolution-complete: it will find a solution if one exists at the level of discretization of the C-space. The path will be a shortest path subject to the allowed motions .

This image below represents a simple map used to apply the A* algorithm in motion planning.The green dot represents the starting position, the blue dot represents the goal, and the black rectangles are obstacles that the robot must avoid. The A* algorithm searches for the shortest possible path from the start to the goal, intelligently and efficiently navigating around the obstacles.

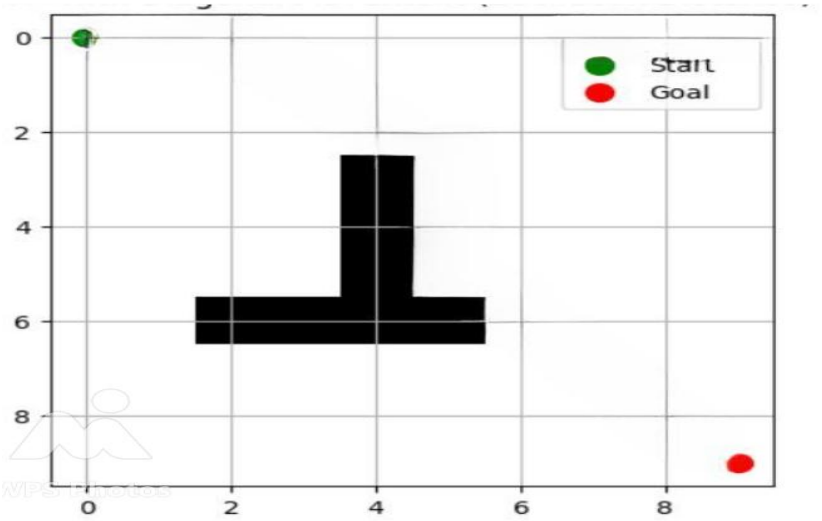

Figure (1) planning problem

Code:

```
# ---------- تشغيل الخوارزمية ----------
path = astar(grid, start, goal)

# ---------- رسم المسار ----------
plt.imshow(grid, cmap='gray_r')
if path:
    path_x, path_y = zip(*path)
    plt.plot(path_y, path_x, color='orange', linewidth=2, label='A* Path')

plt.scatter(start[1], start[0], c='green', s=100, label='Start')
plt.scatter(goal[1], goal[0], c='red', s=100, label='Goal')
plt.legend()
plt.title('A* with Diagonal Movement (Euclidean Distance)')
plt.grid(True)
plt.show()
```

The search result was represented by applying the A* algorithm to find the optimal path from the starting point (in green) to the target (in blue).

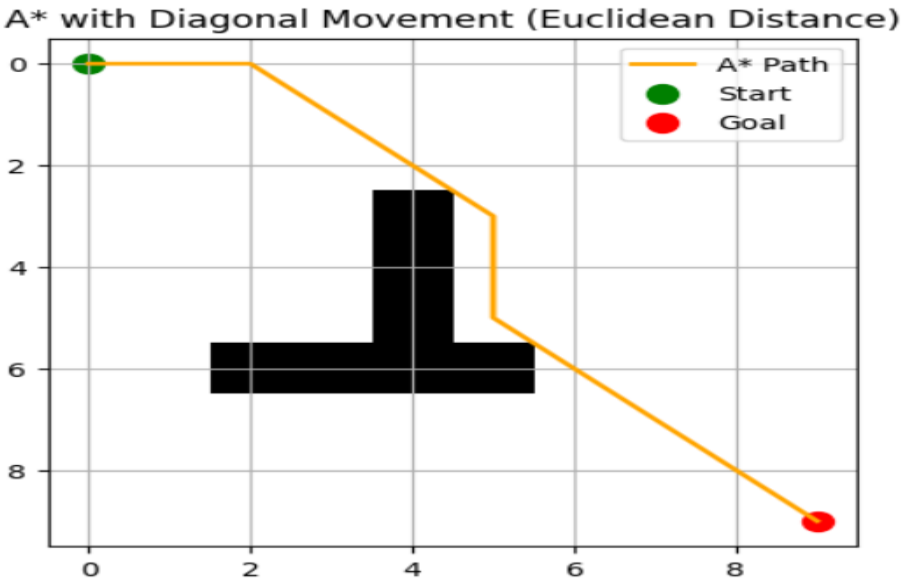

Figure (2) An A* path-finding on grid

The red line shows the path found by the algorithm, which successfully avoided the black obstacles and chose the shortest possible route to reach the target efficiently.
However, this path is often:

- ✓ Jagged (due to the grid)
- ✓ Not smooth (as if moving in square steps)
- ✓ Not necessarily the "best in reality" for the robot

This is where the Flower Pollination Algorithm (FPA) comes in to improve this path generated by A*.

5. Flower Pollination Algorithm (FPA)

The Flower Pollination Algorithm (FPA), proposed by Xin-She Yang (2012), is a population-based metaheuristic algorithm that is inspired by pollination in plants. FPA switches between global and local pollination with a probability p. In this work, the environment is represented as a two-dimensional grid, and each FPA solution is represented as a set of waypoints. In this work, the objectives of the optimization process include minimizing path costs, smoothness penalties, and proximity to obstacles. This is achieved through a nature-inspired search algorithm that is flexible and stochastic in its approach and is used in conjunction with the deterministic approach of A*.

Summary of the Method:

1. The environment is represented on a 2D grid: free cell or obstacle.
2. Each solution in FPA = a list of N_waypoints (each point has real coordinates within the map boundaries; we later round them to the center of the cells).
3. To calculate the cost of a solution:
   - We divide the path into segments between [start, wp1, wp2, ..., goal].
   - For each segment, we calculate a grid path using A*; if A* fails (no path), we assign a very high cost.
   - Solution cost = sum of segment lengths + α * (number of turns/smoothness) + β * (obstacle proximity penalty).
4. FPA optimizes the set of waypoints to minimize the cost.
5. We output the best resulting path (composed of the A* results between the best waypoints).

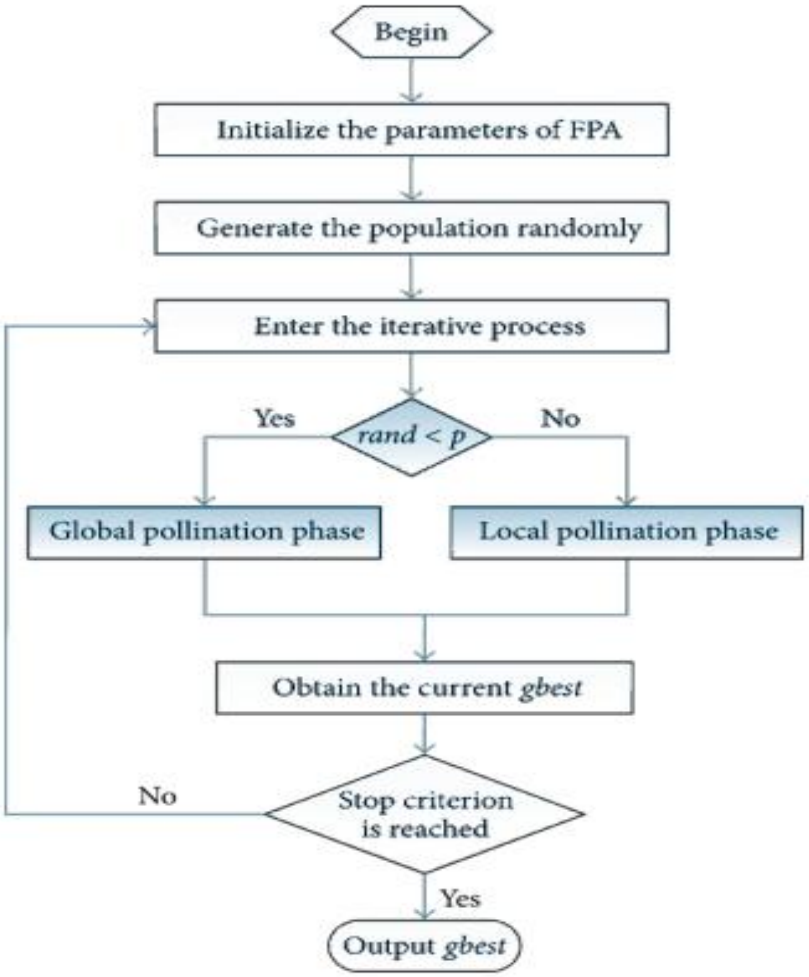

Figure (3) FPA Flowchart.

6. Hybrid A*-FPA Methodology

The Hybrid A*-FPA model integrates the determinate efficiency of A* with the adaptive search of FPA. The Hybrid A*-FPA model, as presented in the literature (Yang, 2012; Lynch & Park, 2019), follows this procedure:

Step 1: The A* algorithm is applied to calculate an initial feasible path on a grid map.

Step 2: The FPA model treats intermediate waypoints as variables and optimizes them using its optimization process.

Step 3: The resulting path has local optimality (from A*), which is integrated with global smoothness (from FPA). The path is smooth and of minimum length, making it feasible and energy-efficient for real robots.

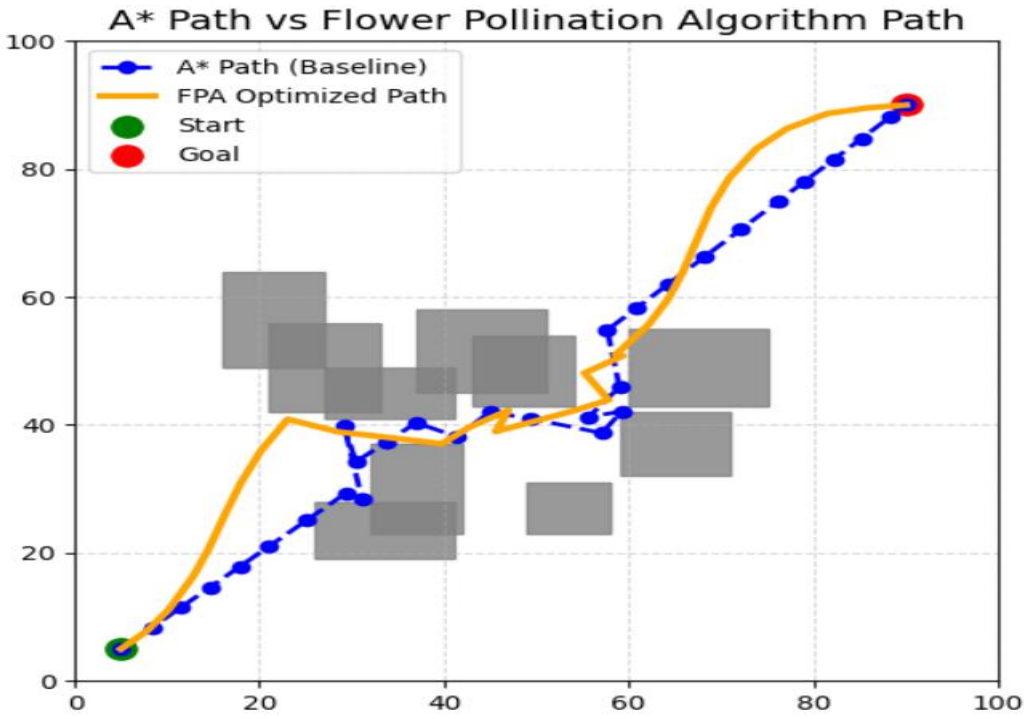

Figure (4) illustrates the result of hybrid traditional A* algorithm and the Flower Pollination Algorithm (FPA) by having them work together on a map containing obstacles (in black) and the latter's effect on the robot's path in navigating around the obstacles.

The main components in the image are:

- ✧ Green square = Start point
- ✧ Red mark = Goal
- ✧ Black squares = Only 10 large, rectangular obstacles are present and clearly visible.
- ✧ Small blue dots = Intermediate points resulting from path optimization by the FPA algorithm
- ✧ Blue line = Path generated by the original A* algorithm
- ✧ Orange line = Optimized path from the A* + FPA algorithm
- ✧ The blue path (A*) is the traditionally optimal path, meaning it is the shortest path from the start to the goal according to the evaluation function f(n) = g(n) + h(n).
- ✧ The orange path (A* + FPA) attempts to improve the path generated by A* by:

a) Removing unnecessary turns.
b) Reducing the number of intermediate nodes.
c) Making the path smoother and more realistic (as a real robot would move).

7.RESULTS AND DISCUSSION

Table (1) This illustrates the difference between the results of the A* algorithm and the Hybrid A* + FPA algorithm, where A* produces a correct but longer and more winding path, while Hybrid A* + FPA reduces the path length and makes it smoother thanks to the improvements of the Flower Pollination Algorithm.

Criterion	A* (Blue Line)	Hybrid A* + FPA(Orange Line) |
Path Length	≈ 1551.7 pixels	≈ 140.4 pixels
Number of Nodes	1079 nodes	112 nodes
Total Angle Change (Smoothness)	≈ 560.2 radians	≈ 24.4 radians
Overall Result	Longer and less smooth path	Shorter and smoother path

Simulation experiments were conducted on grid maps containing rectangular obstacles (Lynch & Park, 2019). Results show that the Hybrid A*-FPA algorithm produces a smoother and shorter path than traditional A*. The visual results confirm that the hybrid algorithm minimizes sharp turns and improves smoothness, resulting in a more realistic and efficient trajectory for robot movement. Moreover, the optimization process reduced the total number of nodes significantly, implying reduced computational load and faster convergence (Yang, 2012).

The training was increased multiple times to demonstrate the algorithm's efficiency in finding smooth .and short paths for the robot's movement from the starting point to the target

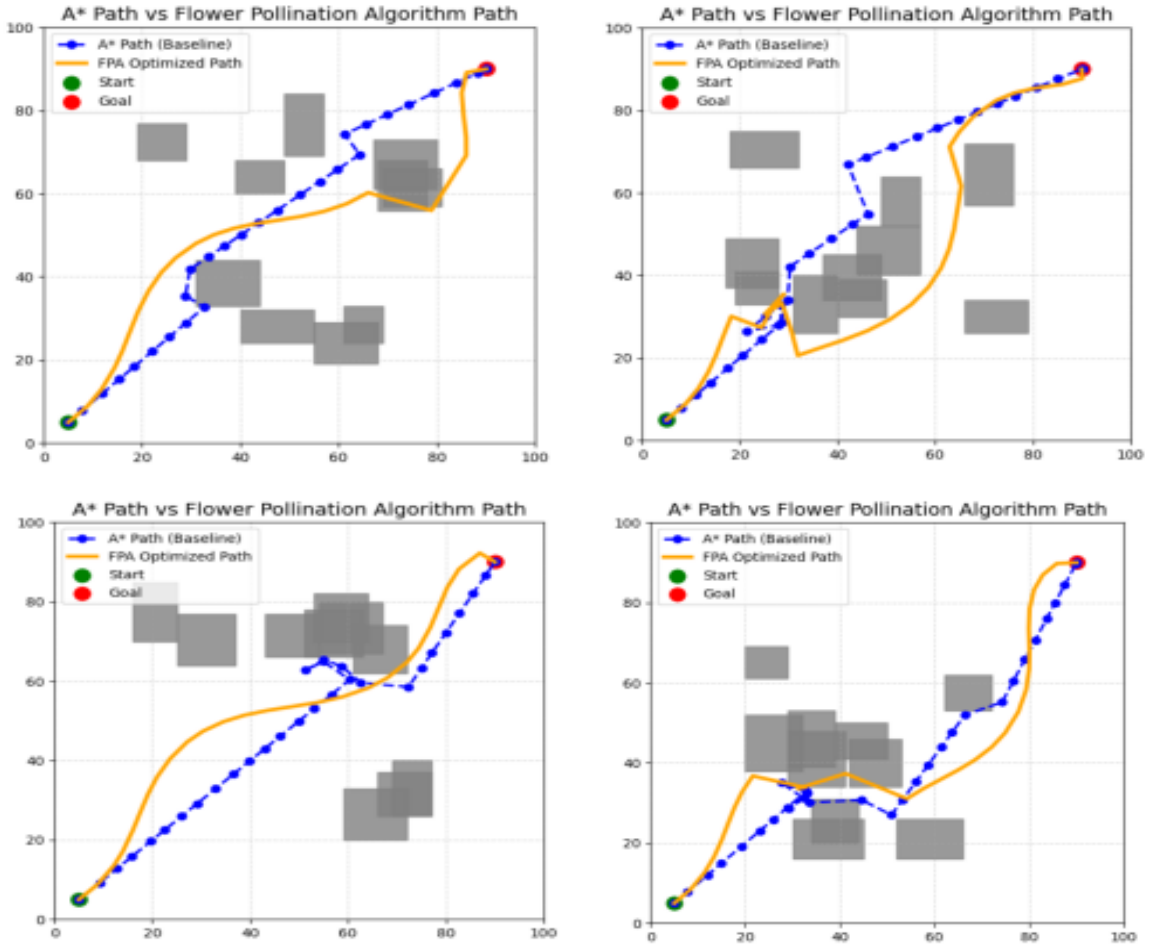

Figure (5)A number of results illustrate the model's training process

Note that the orange (Hybrid) path navigates more smoothly around obstacles compared to the blue path. In some cases, the orange path can actually be shorter or avoid narrow areas. This means the robot will be more stable and consume less energy during movement.

8. CONCLUSION

This study shows that by combining A* with Firefly Performance Algorithm (FPA), it is possible to create more efficient robotic path planning techniques through deterministic and nature-inspired optimization principles. The paths generated by the hybrid algorithm are smoother and more efficient compared to traditional A*. These hybrid methods extend the possibilities of robotic path planning in various environments and may have future uses in autonomous vehicles, search and rescue robotics, and industrial robotics (Yang, 2012; Lynch & Park, 2019).

REFERENCES

1. Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L., & Thrun, S. (2005). Principles of robot motion: Theory, algorithms, and implementations. MIT Press.
2. LaValle, S. M. (2006). Planning algorithms. Cambridge University Press.
3. Lynch, K. M., & Park, F. C. (2017). Modern robotics: Mechanics, planning, and control. Cambridge University Press.
4. Yang, X.-S. (2012). Flower pollination algorithm for global optimization. In International Conference on Unconventional Computation and Natural Computation (pp. 240–249). Springer.
5. Cui, Z., Sun, R., & Wang, Y. (2018). An enhanced flower pollination algorithm for global optimization. Applied Soft Computing, 62, 294–308.
6. Abdel-Basset, M., Mohamed, R., & Sangaiah, A. K. (2018). Metaheuristic algorithms for path planning: A survey. IEEE Access, 6, 67329–67358.

7. Sahar, A., & Hasan, M. (2020). A modified flower pollination algorithm for multi-objective optimization problems. Expert Systems with Applications, 146, 113159.
8. Li, S., & Zhang, Y. (2016). Autonomous robot path planning using the flower pollination algorithm. International Journal of Advanced Robotic Systems, 13(3), 1–10.
9. Mohan, B., & Al-Khazraji, A. (2019). Global path planning optimization for mobile robots using the flower pollination algorithm. Robotics and Autonomous Systems, 112, 29–43.
10. Riaz, M., Noreen, I., & Habib, Z. (2020). Hybrid heuristic–metaheuristic approaches for robot path planning: A review. Applied Intelligence, 50, 2341–2373.
11. Sahu, A., & Padhy, N. (2021). Hybrid A*-based path optimization using metaheuristic search techniques. Engineering Applications of Artificial Intelligence, 100, 104207.
12. Xia, Y., & Zhou, Y. (2017). Genetic algorithm and A* search-based hybrid path planning method. Journal of Intelligent & Robotic Systems, 85, 393–405.
13. Zhang, D., Zhao, D., & Lin, J. (2015). A hybrid A*-PSO algorithm for path planning of mobile robots. International Journal of Control, Automation and Systems, 13(2), 353–362.
14. Alam, A., Reza, M., & Kim, J. (2023). Nature-inspired algorithms for robotics applications: Trends and future directions. Artificial Intelligence Review, 56, 1125–1152.
15. Patel, S., & Xiang, Y. (2022). Metaheuristic optimization techniques for mobile robot navigation: A comprehensive survey. Robotics and Autonomous Systems, 159, 104282.