Journal of Information Hiding and Multimedia Signal Processing ©2025 ISSN 2073-4212
Ubiquitous International Volume **16**, Number **2**, June **2025**

Dynamic Window Approach for Mobile Robot Path Planning based on Hybrid Swarm Algorithms

Walaa Mahmood Hassen[1,*], Saman H. Amin[1]

[1]Computer Engineering Department, University of Technology-Iraq, Baghdad, Iraq
{ce.21.09@grad, 120108}@uotechnology.edu.iq

*Corresponding author: Walaa Mahmood Hassen

Received February 23, 2025, revised May 5, 2025, accepted May 7, 2025.

Abstract. *This paper's goal is to identify the best or almost best course through a complex environment while avoiding collisions with both static and dynamic obstacles. In this study, a dynamic path-planning approach to the mobile robot planning problem in dynamic situations is proposed. The Dynamic Window Approach (DWA) and the hybrid River Formation Dynamic and Particle Swarm Optimization (RFD-PSO) algorithm make up the dynamic path planning method. Firstly, we Initialize particles using the RFD path as the initial position, we Divide the solution space into smaller segments or windows. secondly, using a Sliding Window at each iteration; we apply PSO updates to only a segment (window) of the particle's position and slide the window across the entire solution space over iterations. Thirdly, Update and Evaluate; after updating the positions within the window, we evaluate the fitness of the new positions and update the personal and global bests. Lastly, in Integration with RFD, we Use RFD to handle specific constraints and improve path feasibility by refining the positions obtained from PSO. The comparing simulations between the hybrid (RFD-PSO) algorithm sliding window method, and the Dijkstra Algorithm Dynamic Window Approach, the proposed method shows that the proposed hybrid approach outperforms the Dijkstra Algorithm Dynamic Window Approach method, which travels more efficiently and optimizes paths in dynamic environments.*
Keywords: Mobile robot, Path Planning, Particle Swarm Optimization, River Formation Dynamic, Dynamic Window approach, Dijkstra algorithm, dynamic environment

1. **Introduction.** Robots are used in many fields, possess distinct design characteristics, and are therefore categorized and grouped based on their functions. Researchers universally identify robotics as a distinct field that encompasses the intellectual advancement of manufacturing, building, and several other aspects of human existence [1]. Mobile robots are used in several sectors such as industry, agriculture, space exploration, military operations, medical field, education, rescue missions, and more [2]. A mobile robot must be able to navigate its current location to its destination position, finding a path that avoids both static and moving obstacles. It should also optimize its path to save time and energy, finding the fastest and safest way to reach its destination [3]. To attain secure and independent navigation, the mobile robot must be capable of seeing its surrounding environment and devising a course that is free from collisions, and It should be able to traverse seamlessly in an unfamiliar setting, evading obstacles without coming to a halt [4].

For mobile robots to operate efficiently, path planning technology is essential. Finding the most effective or nearly ideal route from the starting point to the destination while avoiding obstacles in a complicated environment is what it comprises. Predetermined

initial and target locations control the robot's movements [5]. Fixed path planning [6] and dynamic path planning [7] are the two main categories into which robot route planning is typically separated in scholarly discussions.

Contemporary mobile robots often rely on a pre-established magnetic pathway to gather things from a certain location, resulting in a limited selection of routes. When many mobile robots are working concurrently, there is a possibility of different types of accidents or incidents occurring among them. Hence, more investigation is important to examine the act of evading impediments between them. Path planning technology is an essential and vital technology for mobile robots, since it serves as the foundation and need for performing the following actions. By optimizing the robot's route, we may limit the distance it has to go and enhance the smoothness of its trajectory. This optimization can lead to increased efficiency, safer navigation, less mechanical stress on the robot, a longer lifespan, and lower energy consumption. Hence, the investigation of the global route planning issue has significant economic and practical importance [8].

For robots to effectively navigate their paths, path planning is essential. It uses two primary approaches: local route planning, which adapts dynamically to real-time sensor inputs, and global route planning, which depends on pre-existing environmental data. Path-planning strategies find the best route while lowering the chance of collisions by combining obstacle avoidance algorithms with the robot's current position and map data [9]. Local route planning is the process of creating a path that avoids collisions, even when there is little or no knowledge about the surroundings. The sensors on the mobile robot must provide real-time updates on the size, shape, and placement of impediments. Otherwise, acquiring the barrier distribution in the vicinity would be unattainable. Global route planning involves using known environmental information, such as the beginning posture and goal location, to repeatedly compute and determine the ideal collision-free path from the starting node to the target node. The assessment of route quality should be conducted based on certain parameters, including distance, duration, smoothness, and energy. Distance and time are the predominant characteristics among them [10].

Two major issues in mobile robot route planning are collision avoidance and determining the most effective and seamless route from the starting point to the destination. The current work uses a hybrid technique to address these issues. The River Formation Dynamic (RFD) method and Particle Swarm Optimization (PSO) are combined in the suggested hybrid approach, RFD-PSO. The dynamic window technique is used to manage moving barriers and reroute the robot to the destination.

This organization of this paper is as follows: Section 2 describes the hybrid methods. Section 3 describes the proposed dynamic window approach. In Section 4, Simulations are performed, and the outcome is displayed. Section 5 addresses the conclusion.

2. **Algorithms.**

2.1. **River Formation Dynamic Algorithm.**

One metaheuristic method that draws inspiration from nature is the River Formation Dynamics (RFD) algorithm. Its main idea is to replicate the movement of water droplets from higher altitudes to lower ones, which is how riverbeds naturally form. Both erosion, which lowers ground levels, and sedimentation, which increases them, are effects of this process on the environment [11]. The RFD algorithm's steps are described as follows:

```
initializeNodes ()
initializeDrops ()
while (not endConditionMet ())
  moveDrops ()
```

```
  analyzePaths ()
  erodePaths ()
  depositSediments ()
end while
```

Drops are initially dispersed around the surroundings at random. Additionally, nodes are started in two steps: all other nodes are given the same initial altitude, and the target node's height is set to 0 (signaling the sea, which is the final destination for all drops). The algorithm runs in a while loop that keeps going until either an alternative stopping requirement is satisfied or all drops follow the same collection of nodes. If, after n iterations, the optimal solution is still the same, this criterion may restrict the number of iterations, execution duration, or end the loop [12]. the probability that drops k at node i selects the next node j as defined in Eq. (1) is:

$$P(\mu_{i,j}) = \begin{cases} \frac{\nabla_{\mu_{i,j}}}{\sum_{l \in Vk(N(i))} \nabla_{\mu_{i,l}}} & \text{if } j \in (N(i)) \\ 0 & \text{if } j \notin (N(i)) \end{cases} \tag{1}$$

(i) represents the collection of neighboring nodes that a drop may traverse from node i. Diminishing the negative gradient of nodes i and j is represented by the term $decreasingGradient(i, j)$. The equation for the negative gradient is shown in Eq. (2) [13]:

$$\nabla_{\mu_{i,j}} = \frac{altitude(i) - altitude(j)}{distance(\mu_{i,j})} \tag{2}$$

Node x's height is represented by the function $altitude(x)$, while the length of the edge connecting nodes I and J is shown by the function $distance(i, j)$. The altered paths are subjected to an erosion process once every droplet has been moved. A node's height is lowered by this erosion in direct proportion to its gradient with respect to the node next to it. Erosion occurs at node A specifically when a droplet travels from node A to node B. Each node in the graph has a modest rise in height when the erosion process is finished [14].

After many repetitions, the erosion function generates a scenario with an almost negligible height (nearly zero). Consequently, this leads to a minimal gradient and disrupts all established pathways. In the last stage, it is necessary to thoroughly examine all the solutions obtained from the drop and retain the most optimal solution found so far.

The RFD method has various limitations that hinder its performance in the specified route creation issue [15]. Due to the abundance of coefficients, fine-tuning the method for a specific scenario is very counterintuitive. Furthermore, its convergence rate is low when dealing with more complex settings.

2.2. **Particle Swarm Optimization (PSO):.** The optimization method known as Particle Swarm Optimization (PSO) [16, 17] was motivated by the collective behavior seen in social groups such as schools of fish or flocks of birds. PSO seeks to emulate these social animals' behavior, in which there is no central leader and people are able to accomplish their objectives by effectively communicating with one another. This method makes it easier to explore the solution space by representing each "swarm particle" as a possible solution to an optimization problem. The particles are a collection of potential solutions that PSO looks for in the specified area. The dynamic velocity of each particle dictates how far it can travel in the search space. Each particle also possesses memory, which allows it to save the best answer it finds while exploring the space [18].

The method may be succinctly outlined in four primary stages, which are iterated until the termination criterion is met as follows [19]: (1) Allocate initial random locations and

velocities to all particles inside the search space.
(2) Assess the level of fitness for each particle.
(3) Modify both the individual and global best.
(4) Modify the velocity and position of every particle.

The Eq.3 (3) and Eq. (4) [19] are used to determine the velocity and position update of particle i at the $t+1$ iteration.

$$Vi(t+1) = \omega * Vi(t) + C1 * R1 * (Pbest - Xt(i)) + C2 * R2 * (Gbest - Xi(t)) \quad (3)$$

$$Xi(t+1) = Xi(t) + Vi(t+1) \quad (4)$$

The inertia weight is represented by the symbol ω. The self-learning component, denoted by $C1$, is the weight coefficient for learning from the historical best location of a particle. As the weight coefficient for learning from the global best position, $C2$ is the global learning factor. Values between 0 and 1 are produced at random for $R1$ and $R2$. Particle I's historical best position is called Pbest, while its present global best position is called Gbest. $Vi(t)$ represents the particle i's velocity at time t, while $Xi(t)$ indicates its position at that time.

Although the PSO method is proficient at properly solving the route planning problem and generating a seamless path, it often encounters the issue of rapidly converging to local optima in various optimization issues. Furthermore, the convergence rate for multidimensional problems is frequently unpredictable, and there is no generally applicable convergence criterion for Particle Swarm Optimization (PSO) in real-world applications [20]. Furthermore, in complex cases, this algorithm does not ensure an ideal answer.

2.3. **Hybrid Method:** Finding the optimal route between the beginning node and the destination node is the primary objective of this approach. The RFD and PSO algorithms serve as the foundation for the suggested hybrid path-planning method. The RFD-PSO algorithm is a hybrid algorithm created in this thesis by combining two methods. The suggested hybrid RFD-PSO method produces a low path distance, a smooth path, and no collisions with barriers in settings with stationary and moving obstacles.

The River Formation Dynamic algorithm can find an optimal path, but sometimes it gives a sub-optimal path because the algorithm depends on the gradient. When rivers pass through semi-flat areas, this leads to a sinuosity in the path, and thus a sinuosity in the path, the path is relatively long, and the gradient is low.

Also, the particle swarm optimization algorithm is easy to fall into the local optimum and it produces an ineffective solution in the narrow paths. A hybrid algorithm (RFD-PSO) that combines the two methods is used to find the shortest path with collision avoidance.

By directing the mobile robot toward its objective while dodging obstacles, the River Formation Dynamics (RFD) algorithm makes sure that a path is found from the starting position to the target point. Following the robot's arrival at the target, these links are strengthened using the Particle Swarm Optimization (PSO) method, which optimizes the route between every pair of important sites. The most effective route from the starting point to the destination is thus produced by the algorithm.

This technique takes a hybrid approach, with one of the particles in the PSO algorithm starting at the RFD-determined first predicted path to the target. The PSO method iteratively optimizes each particle, which represents a possible path the robot could take. Both the individual best position of each particle and the global best position attained by the entire swarm are used to update particle positions. Velocity restrictions are used to avoid too much movement in a single repetition. A particle's velocity is inverted if its new position is outside of the predetermined range. The path produced by the RFD

technique serves as the initial guidance for the PSO algorithm, ultimately determines the robot's final path based on the global best solution it finds. Figure (1) illustrates the steps of the suggested hybrid RFD-PSO algorithm. The red solid line indicates the optimal path direction from the robot's starting point (left) to the target point (right), dynamically avoiding obstacles. A flowchart that shows the mobile robot's best path is shown in Figure (2).

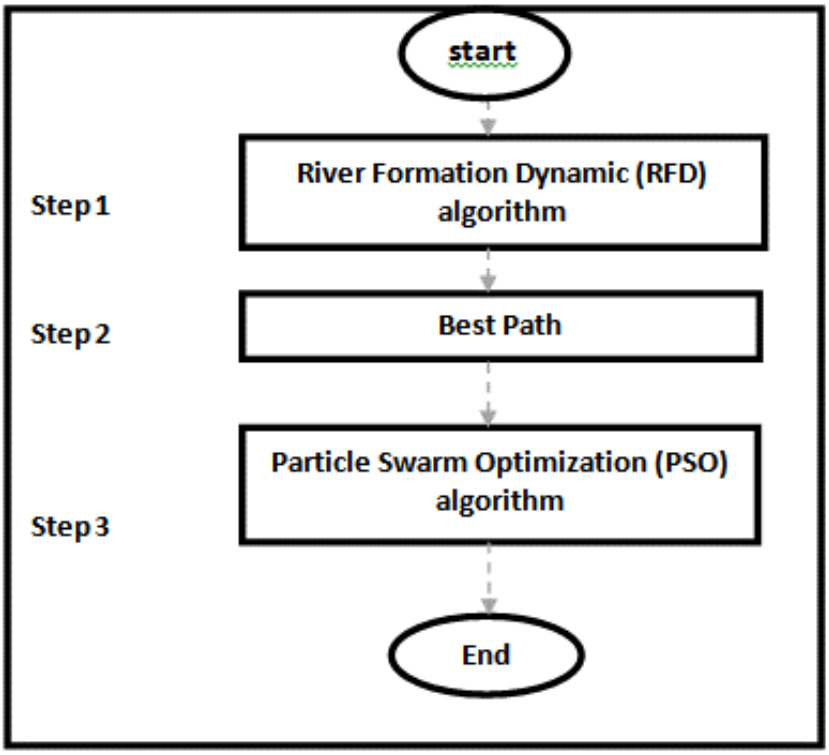

FIGURE 1. Steps of Hybrid RFD-PSO algorithm

FIGURE .2 Flowchart of the RFD-PSO hybrid algorithm

3. Sliding window approach and its application in path planning.

3.1. **Algorithms and Implementations.** The route design process may use the Sliding Window Approach, which involves breaking the way into smaller sections or windows and assessing the viability of each window individually. This strategy is especially beneficial when dealing with intricate situations, as the route may be improved by simultaneously examining many windows. The following algorithms use the sliding window technique for route planning [21]: (1) The Sliding Window Informed RRT method is a technique that enhances the efficiency of route planning by integrating the Sliding Window Approach with the Rapidly exploring Random Tree (RRT) algorithm [22]. The method keeps track of a moving window that represents the current route and utilizes it to guide the RRT algorithm's exploration for a viable path.
(2) Enhanced Sliding Window method: This method aims to enhance the effectiveness of the Sliding Window Approach by optimizing the size of the window and adapting the sliding window according to the current route [23]. The algorithm's objective is to decrease the quantity of windows required to cover the complete route, hence decreasing the time complexity of the path planning process.
(3) Path Planning using Sliding Window and Variant A: This approach integrates the

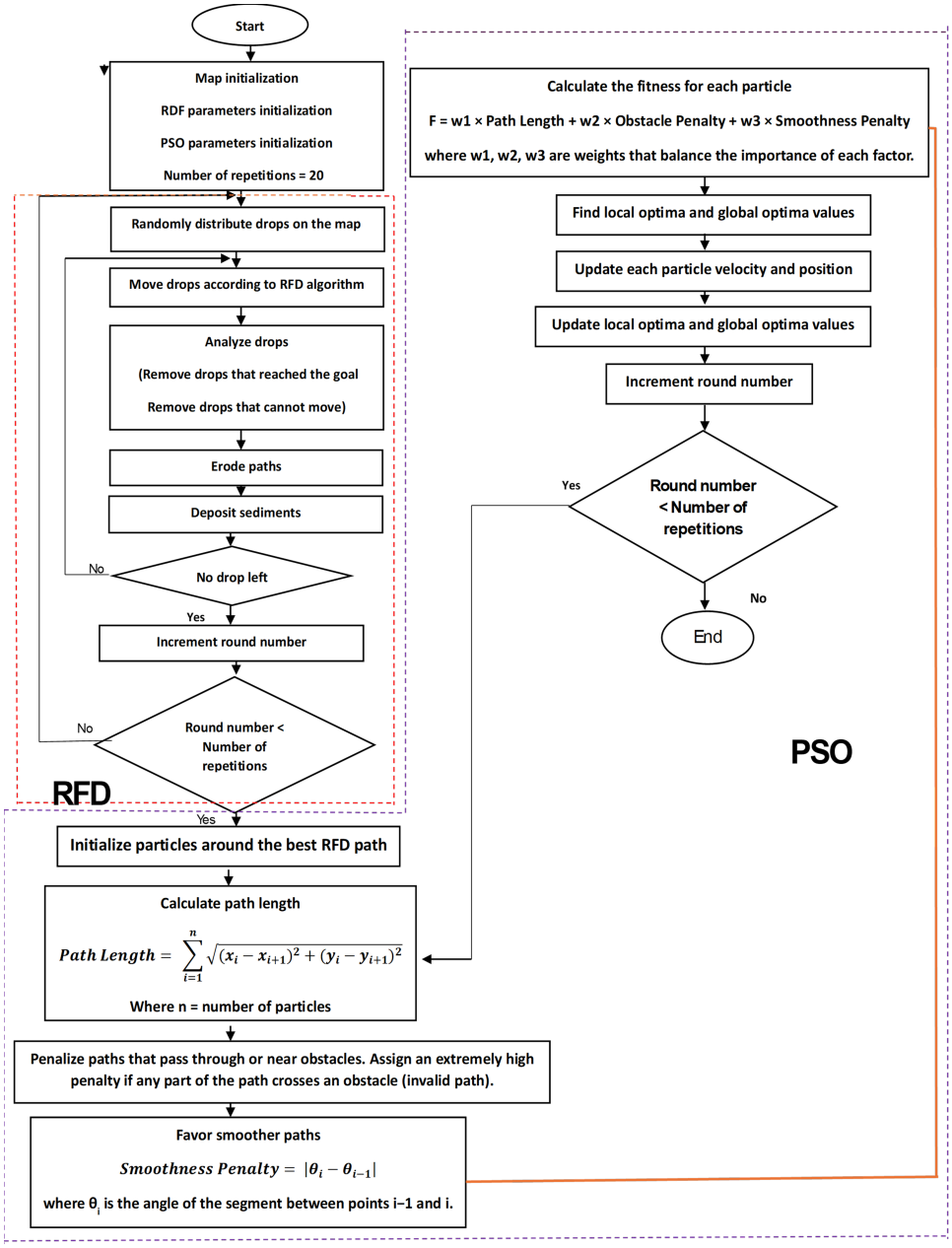

FIGURE 2. Flowchart of the RFD-PSO hybrid algorithm

Sliding Window Approach with the A* algorithm, a well-known pathfinding technique, to enhance the efficiency of the path planning process [24]. The technique uses a sliding window to assess the viability of various route segments and leverages the A* algorithm to determine the most optimal path via the windows.

3.2. **Sliding Window Approach in Path Planning.** The Sliding Window Approach is a method used in route planning for autonomous systems, such as robots and drones. The process entails partitioning the trajectory into smaller segments (windows) and assessing the viability of each window. This strategy is especially beneficial when dealing with intricate environments, as the route may be improved by simultaneously examining many windows [11].

In Figure (3), two different scenarios showing the movement of obstacles to prevent entering hazard regions are introduced. The hazard region is shown in Figure (3(a)), with

an obstacle's initial and final positions featured according to its directionality. Figure (3, b) expands on this by detailing the distances relative to the obstacle's initial position. This provides a clear view of how the dynamic obstacle navigates around the hazard region. These representations are essential for understanding the spatial dynamics and planning necessary for effective obstacle avoidance in dynamic environments.

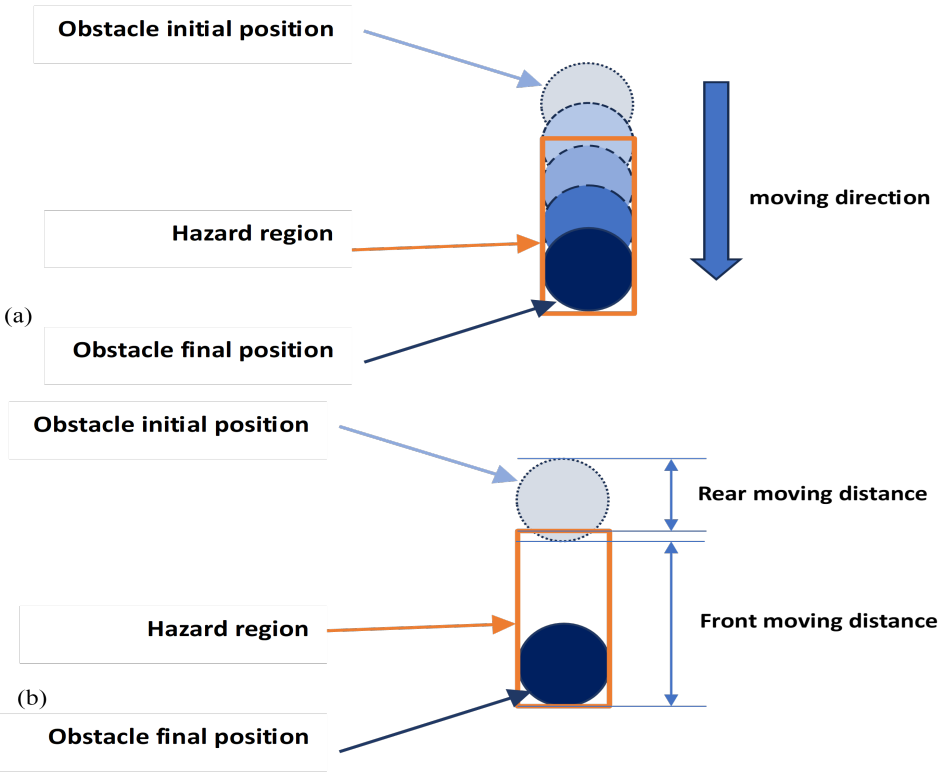

FIGURE 3. Moving Obstacle Representation to Avoid Hazard Region. (a) Hazard Region Representation, (b) Distances with Respect to Initial Obstacle Position

Figure (4) illustrates a schematic of how to compute time as a function of the shortest distance from the robot and total path length. The illustration contains a sliding window, a robot (red), an obstacle, and the original path of the Robot. This schematic shows that a real-world robot needs to consider the distance of obstacle and path length, which will decide how quickly it can move around its space efficiently without hitting obstacles. The present work is crucial when performing real-time path planning and collision avoidance in robotics.

Figure (5) depicts a slower obstacle moving at constant velocity. It shows the initial path of the robot, and a new path must be selected through a feathered area where it came in that hazard region because of the obstacle. This figure shows the decision-making paradigm of path selection in moving obstacles that reveals how adaptive strategies are necessary to maintain this robot moving safely and continuously.

Figure (6) addresses a similar scenario, but it involves a fast-moving obstacle this time. The Figure illustrates the initial path to avoid obstacles and where it deviates due to a higher velocity of the new object. This scenario illustrates the increased complexity in path planning required to accommodate faster obstacles, underscoring the necessity for rapid and efficient algorithmic responses to maintain collision-free navigation.

Figure (7) shows the scenario of moving obstacles with different speeds, indicating the requirement to rescale the dynamic sliding window. This figure shows the various

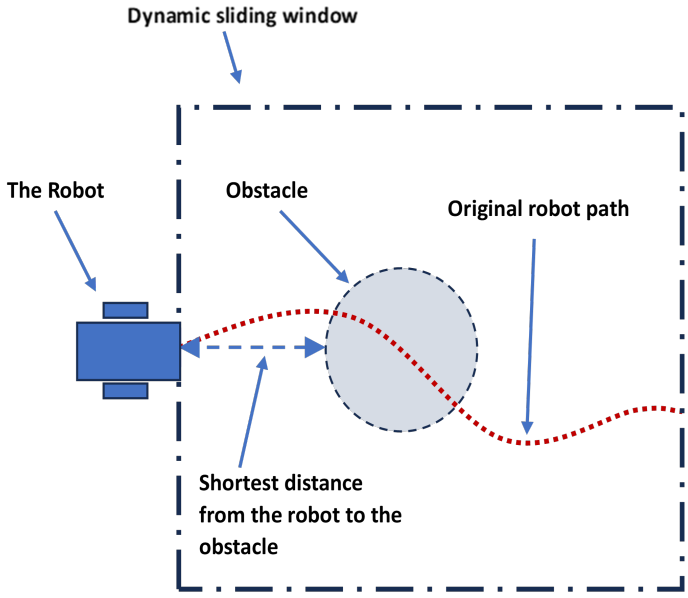

FIGURE 4. Time Calculation based on Shortest Distance to Obstacle and Path Length

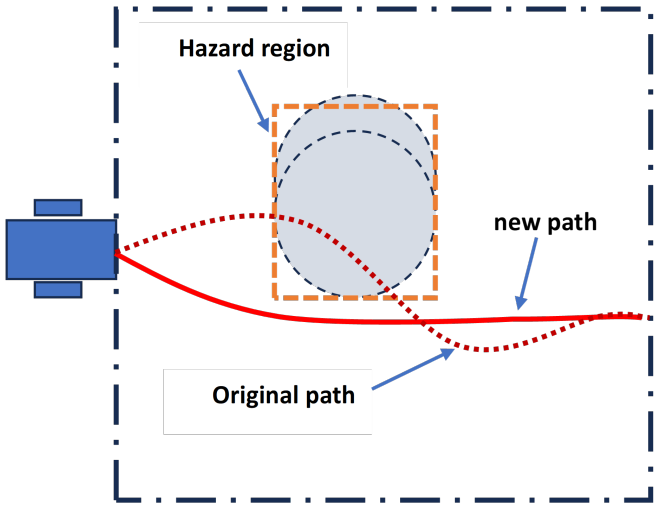

FIGURE 5. Slow Moving Obstacle with Constant Speed and Probable Path Selection

computations, such as for the rear and front-end moving distances (based on minimum v max or maximum speed). This dynamic tuning is essential for accurately predicting the obstacles' future locations and allowing safe navigation around hazard regions. Being able to resize the window for varying speed of obstacles makes the proposed path-planning algorithm versatile and robust.

3.3. **Sliding Window Approach in Hybrid RFD-PSO Algorithms.** The sliding window technique may be included in a hybrid RFD-PSO algorithm in the following manner:

(1) Initialization: The particles are initialized by setting their starting location to the

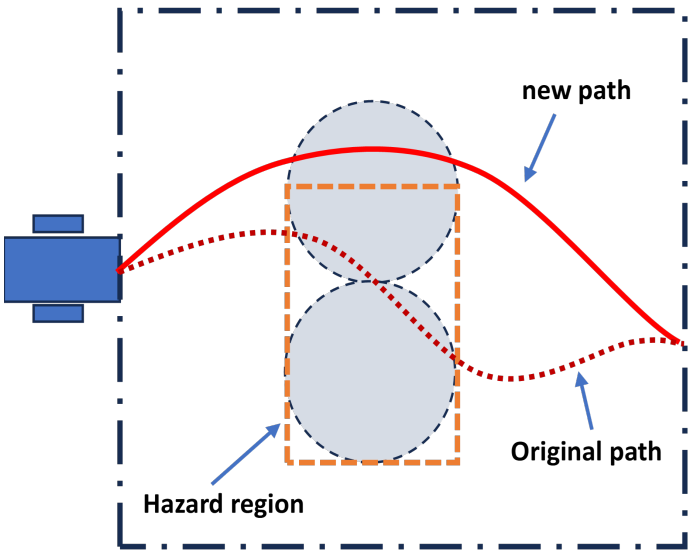

FIGURE 6. Fast Moving Obstacle with Constant Speed and Probable Path Selection

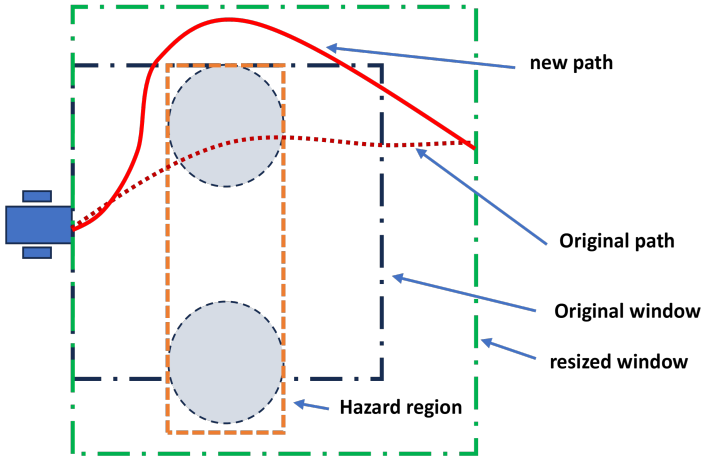

FIGURE 7. Moving Obstacle with Variable Speed Showing the Need to Resize the Dynamic Sliding Window, (minimum speed is zero and used with the calculation of rear end moving distance, maximum speed is known and is high which is used to calculate the front-end moving distance).

RFD route. Partition the solution area into smaller parts or windows.
(2) The Sliding Window technique allows for applying Particle Swarm Optimization (PSO) at each iteration, and updates to a specific segment or window of the particle's location. This window may be moved throughout the whole solution space over iterations.
(3) Update and Evaluate: Following the update of positions inside the window, we assess the effectiveness of the new positions and modify the personal and global bests accordingly.
(4) Integration with RFD involves using RFD to address certain limitations and enhance

the feasibility of a route by refining the positions acquired by PSO.

The flowchart in Figure (8) outlines the procedural steps a robot follows to travel from its starting point to its destination. The flowchart starts with the robot detecting an obstacle and setting a dynamic sliding window. From there, the robot gradually moves towards locating the hazard region or an obstacle, computing the least distance to reach the obstacle and identifying feasible paths. This process includes resizing the window size and then computing the total time to pass through that, which, if it leads to only inside a hazard region, then the robot must change its path. This detailed sequence emphasizes the systematic approach required for effective robot navigation in dynamic and obstacle-rich environments.

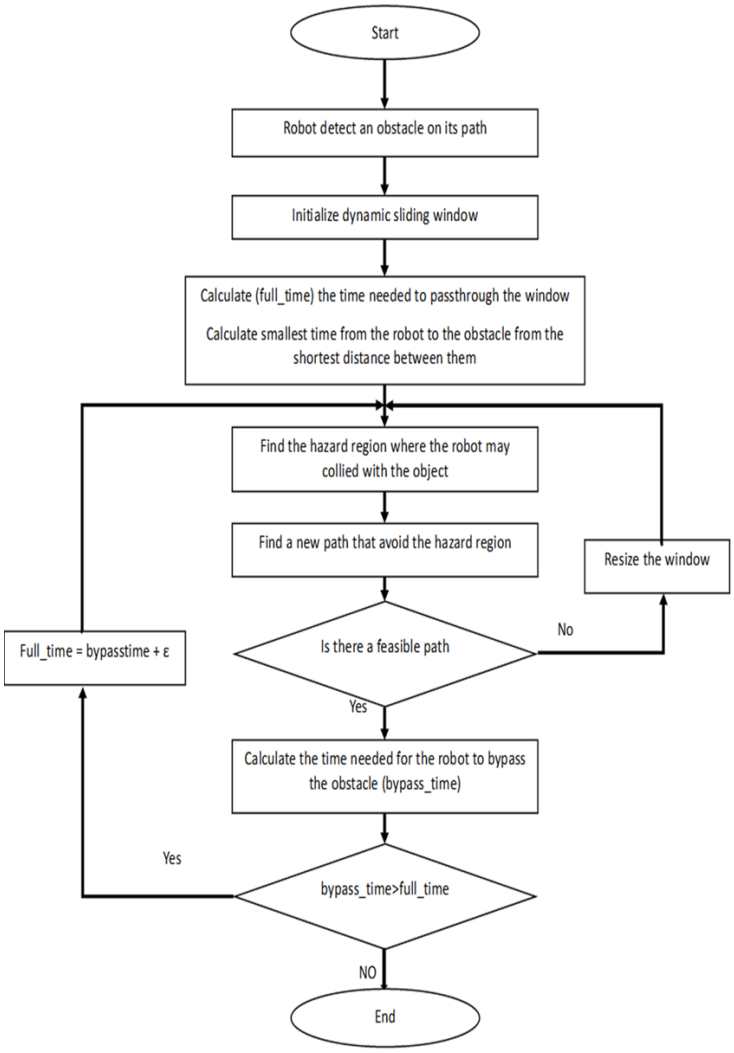

FIGURE 8. Flowchart of the General Sliding Window Approach

4. **Methodology.** In this paper, two types of hybrid approaches to robot path planning will be studied under dynamic and obstacle environments, and a performance comparison will be conducted. The key focus will be to check the path length the robot has covered from the starting point to the destination point. The performance of the proposed approaches will be evaluated through two cases.

4.1. **Case No.1.** For robot path planning in dynamic, obstacle-filled situations, we employed two hybrid algorithms in this instance. As seen in Figure (9), the algorithms are Hybrid PSO-RFD and Sliding Window applied to Map No. 1. Map No. 1 offers an appropriate test for assessing the algorithms' performance since it depicts a dynamic environment with both static and dynamic obstacles.To measure and compare the path lengths that the robot travels from the beginning point to the destination point while avoiding the dynamic and static obstacles in the environment, both algorithms are run on Map No. 1. To provide a pertinent comparison and assess how well the sliding window method may be included into the hybrid approach, the study uses the same map for both algorithms.

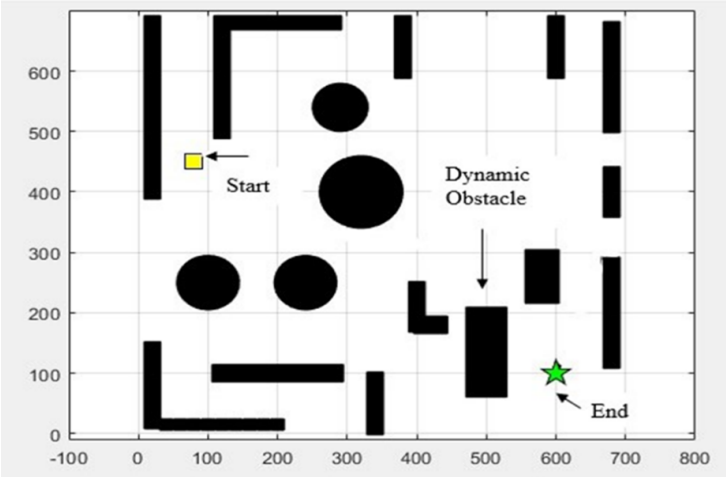

FIGURE 9. Map no.1 with Static and Dynamic Obstacle

Figure (9) illustrates Map 1, a simulated 20×20 m dynamic environment containing each static and moving barriers. Static limitations are represented as black rectangles, whilst dynamic obstacles are depicted as pink circles. Each dynamic obstacle is annotated with a directional arrow (e.G., clockwise or counterclockwise) and its pace (e.G., 0.5 m/s), explicitly indicating motion patterns. The robotic's beginning function (2,2) and goal (18,18) are marked to contextualize navigation demanding situations. These speed annotations align with the experimental parameters defined in Section 4.3, providing readability on how the hybrid RFD-PSO algorithm adapts to limitations with various speeds. The visible cues enhance reproducibility and directly correlate with performance metrics together with route period and collision avoidance efficiency.

4.2. **Case No.2.** In Case 2, the performance of the suggested hybrid algorithms will be evaluated and illustrated by contrasting them with other current techniques from the literature [25] that make use of the Dijkstra Algorithm and the Dynamic Window approach. Map2 (Figure (10)), modified from [25], which incorporates both static and dynamic barriers, will be used for the comparison. This comparison will make it possible to assess how well the suggested hybrid approach performs in comparison to the most advanced methods available today.

Figure (10) showcases Map 2, any other 20×20 m takes a look at surroundings presenting a distinct arrangement of static obstacles (black rectangles) and dynamic limitations (crimson circles). Dynamic boundaries on this map exhibit a broader velocity range (0.3–1.0 m/s), annotated along directional arrows to reflect their movement trajectories. Unlike Map 1, Map 2 includes denser static boundaries and a better number of dynamic

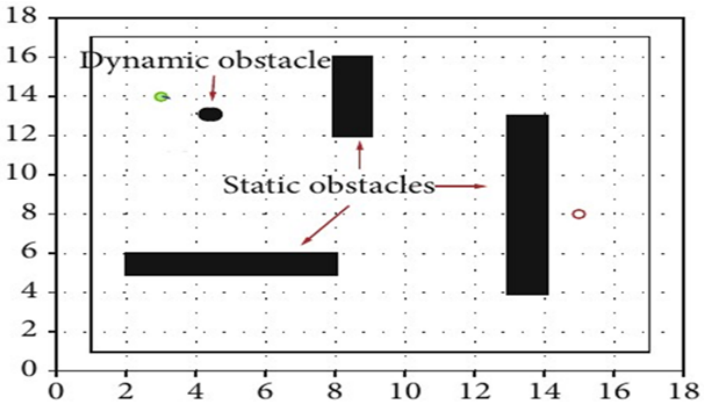

FIGURE 10. Map 2 with Static and Dynamic Obstacles

boundaries, trying out the set of rules's robustness in complex scenarios. The robotic's begin and goal positions are in addition marked, emphasizing the want for adaptive direction making plans. These annotations directly link to the methodology in Section four.Three, allowing readers to visualize how the sliding window resizing mechanism accommodates limitations with diverse speeds, as mentioned within the Results (Section 5).

4.3. **Experimental Environment and Parameter Settings.** The proposed algorithm was evaluated in a **2D dynamic environment** simulated using MATLAB R2023a on a workstation with an Intel i7-12700K CPU (5.0 GHz) and 32GB RAM. The environment comprised:

- **Map Dimensions**: 20 m × 20 m for both Map 1 and Map 2.
- **Static Obstacles**: 5–10 randomly placed rectangular/circular obstacles with fixed positions.
- **Dynamic Obstacles**: 2–4 circular obstacles moving at velocities of 0.3 m/s (slow) to 1.0 m/s (fast).
- **Robot Specifications**: Circular footprint (radius = 0.5 m), maximum linear velocity = 1.2 m/s, acceleration limit = 0.3 m/s^2.

Algorithm Parameters:

- **RFD:**
  - Initial node altitude = 100 units (target node = 0).
  - Erosion rate = 0.1, sedimentation rate = 0.05.
  - Gradient threshold for path feasibility = 0.01.
- **PSO:**
  - Swarm size = 50 particles.
  - Inertia weight (ω) = 0.7, cognitive/social coefficients (C1,C2) = 1.5.
  - Velocity limits = ±0.5 m/s.
  - Maximum iterations = 200.
- **Sliding Window:**
  - Initial window size = 5% of the solution space.
  - Adaptive resizing: ±15% for obstacles with speeds >0.7 m/s.

Stopping Criteria:

- Path convergence: <0.01 m improvement over 20 iterations.
- Maximum computation time per trial = 5 seconds.

5. **Results and Discussion.** In Figure 11, the Dynamic obstacle moves up in the environment; Figure 12 shows the robot's navigation of a new path around the obstacle. These results graphically illustrate that the hybrid algorithms can adapt and maneuver in a dynamic environment.

Table 1 illustrates the best path length score for the proposed hybrid RFD-PSO algorithm and dynamic window approach after skipping the moving obstacle and the time path length after and before moving the dynamic obstacle.

As for Fig. 13, the performance of the proposed hybrid algorithms was further validated by comparing them with well-known approaches in the literature, i.e., Dijkstra Algorithm and Dynamic Window Approach on Map no. 2. In this map, we applied the Hybrid PSO + RFD algorithm and the results are shown in Figure 14.

The suggested hybrid technique considerably shortens the path length when compared to both the Dynamic Window technique method and the Dijkstra Algorithm from earlier studies, as shown by the comparison of the optimal path lengths in Table 2 [25]. In particular, the hybrid approach produces a path length of 14.5568 meters, which is significantly less than the 18.67 meters documented in the literature. This comparison demonstrates the suggested hybrid approach's increased efficacy and efficiency in path optimization in dynamic contexts.

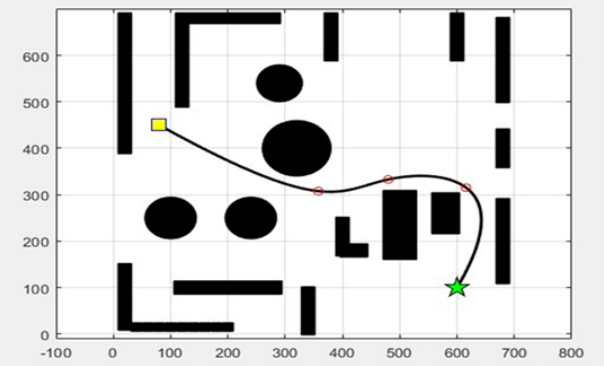

FIGURE 11. The Dynamic Obstacle Moving Up

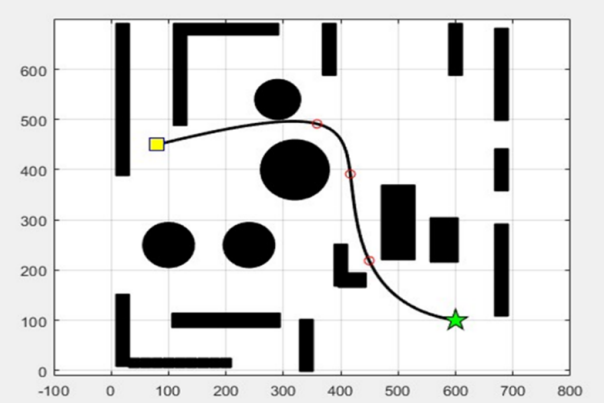

FIGURE 12. The Robot Finds a New Path to Overcome the Dynamic Obstacle

TABLE 1. Best score and Execution Time of the path length

Algorithm	Best score of path length	Execution Time
The Proposed Hybrid and Dynamic Window Approach	606.19 cm	1.26591 s

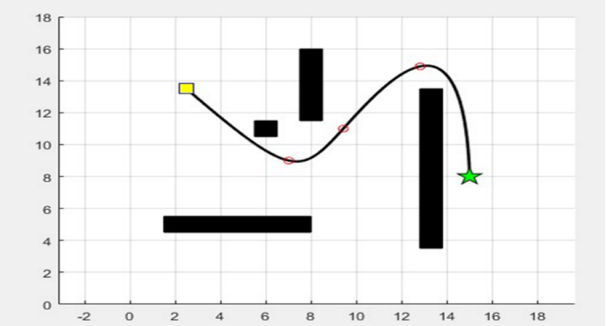

FIGURE 13. The Hybrid Algorithm with Dynamic Obstacle Based on Map 2

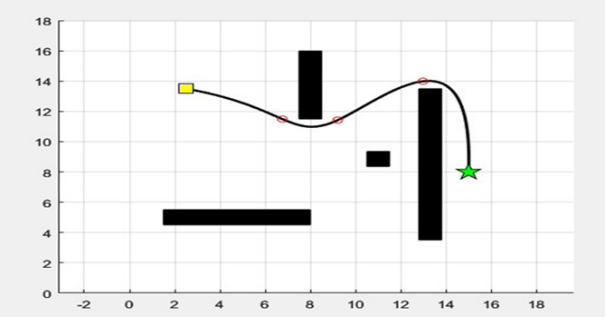

FIGURE 14. The Results for the Proposed Hybrid Algorithm Based on Map 2

TABLE 2. Comparison between the proposed approach and the Dijkstra algorithm and dynamic Window Approach

Algorithm	Path length
Dijkstra Algorithm and Dynamic Window Approach [25]	18.67 m
The Proposed Hybrid Approach	14.5568

6. **Conclusion.** This research addresses two critical demanding situations in autonomous mobile robot navigation—efficient route planning and collision avoidance in dynamic environments—via the development of the hybrid RFD-PSO algorithm. By synergizing River Formation Dynamics (RFD), which mimics natural water float to keep away from limitations thru gradient-based totally navigation, and Particle Swarm Optimization (PSO), which leverages swarm intelligence to refine route smoothness and optimality, the proposed technique generates collision-loose trajectories while minimizing path length. The integration of a dynamic Sliding Window Approach in addition enhances computational efficiency by way of segmenting the answer area and adaptively resizing windows to accommodate limitations with varying velocities (0.3–1.0 m/s). Experimental validation in 20×20 m simulated environments verified a 22.03% discount in path period and an execution time of 1.26 seconds, outperforming traditional techniques just like the Dijkstra-Dynamic Window Approach.

However, the approach faces barriers in extraordinarily restrained environments (>15 static boundaries), where adaptive window resizing struggles to hold optimality, and in multi-robot scenarios, where uncoordinated navigation risks deadlocks. Additionally, actual-time performance degrades with ultra-rapid barriers (>2.0 m/s). Future studies need to extend the framework to 3D environments (e.g., aerial drones), comprise decentralized coordination mechanisms for robot swarms, and integrate gadget mastering for dynamic parameter tuning. Hardware-in-the-loop validation with physically robots and sensors (LiDAR/RGB-D) is likewise vital to assess robustness under real-international noise and latency. These improvements should increase the algorithm's applicability to commercial automation, independent motors, and collaborative robotic systems, addressing both scalability and real-time adaptability.

REFERENCES

[1] T. A. Salih and M. Z. Nayef, "New Design of Mobile Robot Path Planning with Randomly Moving Obstacles," *Tikrit Journal of Engineering Sciences*, vol. 20, no. 1, pp. 21–28, 2013.

[2] B. K. Patle, G. Babu L, A. Pandey, D. R. K. Parhi, and A. Jagadeesh, "A review: On path planning strategies for navigation of mobile robot," *Defence Technology*, vol. 15, no. 4, pp. 582–606, 2019.

[3] I. A. Hassan, I. A. Abed, and W. A. Al-Hussaibi, "Navigation and Path Planning of Mobile Robots," *Journal Européen des Systèmes Automatisés*, vol. 56, no. 5, pp. 751–756, 2023.

[4] Y. M. Al-Sharo, A. T. Abu-Jassar, S. Sotnik, and V. Lyashenko, "Generalized Procedure for Determining the Collision-Free Trajectory for a Robotic Arm," *Tikrit Journal of Engineering Sciences*, vol. 30, no. 2, pp. 142–151, 2023.

[5] P. Sudhakara, V. Ganapathy, B. Priyadharshini, and K. Sundaran, "Obstacle Avoidance and Navigation Planning of a Wheeled Mobile Robot using Amended Artificial Potential Field Method," *Procedia Computer Science*, vol. 133, pp. 998–1004, 2018.

[6] T.-W. Sung, B. Zhao, X. Zhang, C.-Y. Lee, and Q. Fang, "An Efficient Quasi-Affine Transformation Evolutionary Algorithm with Fixed Dimension Updating and Its Application in UAV 3D Path Planning," *Journal of Intelligent & Fuzzy Systems*, vol. 46, no. 4, pp. 9755–9781, 2024.

[7] S. Abdallaoui, E. H. Aglzim, A. Chaibet, and A. Kribèche, "Thorough Review Analysis of Safe Control of Autonomous Vehicles: Path Planning and Navigation Techniques," *Energies*, vol. 15, no. 4, p. 1358, 2022.

[8] Y. Li, J. Zhao, Z. Chen, G. Xiong, and S. Liu, "A Robot Path Planning Method Based on Improved Genetic Algorithm and Improved Dynamic Window Approach," *Sustainability*, vol. 15, no. 5, p. 4656, 2023.

[9] Y. Mhanni and Y. Lagmich, "Enhanced Obstacle Avoidance and Intelligent Navigation for Mobile Robots: An Integrated Approach Using Fuzzy Logic and an Optimized APF Method," *Mathematical Modelling of Engineering Problems*, vol. 10, no. 6, pp. 2111–2120, 2023.

[10] L. Bai and C. Du, "Design and Simulation of a Collision-Free Path Planning Algorithm for Mobile Robots Based on Improved Ant Colony Optimization," *Ingénierie des Systèmes d'Information*, vol. 24, no. 3, pp. 331–336, 2019.

[11] P. Rabanal, I. Rodríguez, and F. Rubio, "Using River Formation Dynamics to Design Heuristic Algorithms," in *Lecture Notes in Computer Science*, vol. 4618, pp. 163–177, 2007.

[12] T.-W. Sung, L. Sun, and K.-C. Chang, "Multi-hop Route Planning Based on Environment Information for Path-Following UAVs," in *Advances in Intelligent Systems and Computing*, vol. 1153, pp. 831–839, 2020.

[13] G. Redlarski, M. Dabkowski, and A. Palkowski, "Generating Optimal Paths in Dynamic Environments Using River Formation Dynamics Algorithm," *Journal of Computational Science*, vol. 20, pp. 8–16, 2017.

[14] A. Cahya Kemila and W. Fawwaz Al Maki, "Parameter Optimization of Support Vector Machine Using River Formation Dynamic on Brain Tumor Classification," *Journal of Electronics, Electromedical Engineering & Medical Informatics*, vol. 5, no. 3, pp. 177–184, 2023.

[15] G. Redlarski, A. Pałkowski, and M. Dabkowski, "Using River Formation Dynamics Algorithm in Mobile Robot Navigation," *Solid State Phenomena*, vol. 198, pp. 138–143, 2013.

[16] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, pp. 1942–1948.

[17] O. A. R. A. Wahhab and A. S. Al-Araji, "Path Planning and Control Strategy Design for Mobile Robot Based on Hybrid Swarm Optimization Algorithm," *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 3, pp. 565–579, 2021.

[18] G. W. Abedulabbas and F. R. Yaseen, "Design a PI Controller Based on PSO and GWO for a Brushless DC Motor," *Journal Européen des Systèmes Automatisés*, vol. 55, no. 3, pp. 331–338, 2022.

[19] Z. Yang, N. Li, Y. Zhang, and J. Li, "Mobile Robot Path Planning Based on Improved Particle Swarm Optimization and Improved Dynamic Window Approach," *Journal of Robotics*, vol. 2023, no. 1, pp. 1–16, 2023.

[20] S. K. Debnath *et al.*, "Different Cell Decomposition Path Planning Methods for Unmanned Air Vehicles—A Review," in *Proc. 11th National Technical Seminar on Unmanned System Technology*, pp. 99–111, 2021.

[21] A. Epasto, M. Mahdian, V. Mirrokni, and P. Zhong, "Improved Sliding Window Algorithms for Clustering and Coverage via Bucketing-Based Sketches," in *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 3005–3042, 2022.

[22] H. Zhang, X. Rong, Y. Li, B. Li, J. Zhang, and Q. Zhang, "Path Planning Based on Sliding Window and Variant A* Algorithm for Quadruped Robot," *High Technology Letters*, vol. 22, no. 3, pp. 334–342, 2016.

[23] H. Y. Zhang, W. M. Lin, and A. X. Chen, "Path Planning for the Mobile Robot: A Review," *Symmetry*, vol. 10, no. 10, 2018.

[24] J. R. Sánchez-Ibáñez, C. J. Pérez-Del-Pulgar, and A. García-Cerezo, "Path Planning for Autonomous Mobile Robots: A Review," *Sensors*, vol. 21, no. 23, 2021.

[25] L. S. Liu *et al.*, "Path Planning for Smart Car Based on Dijkstra Algorithm and Dynamic Window Approach," *Wireless Communications and Mobile Computing*, vol. 2021, pp. 1–12, 2021.