Optimizing Motion in Nanoscale Robotics: A Hybrid WQPSO-Fuzzy Logic Approach for Dynamic Path Planning

Praveen MURUGAIAH[1]*, Dhandapani SAMIAPPAN[1], Bharathiraja NAGU[2]

[1] Department of Electronics and Communication Engineering, Saveetha Engineering College, Chennai 602105, Tamil Nadu, India
praveenmuru85@gmail.com (*Corresponding Author*), dhandapani.me@gmail.com

[2] Department of Computer Science and Engineering, School of Engineering, Dayananda Sagar University, Bangalore 562 112, Karnataka, India
rajamesoft@gmail.com

Abstract: Path planning is crucial for robotics, particularly in navigating dynamic environments with moving obstacles, a challenge that remains only partially addressed in the existing literature. This study aims to improve the movement of nanorobots by tackling the challenges related to the planning of the optimal routes for navigating the common paths used in factory settings. The analyzed model sets important control settings and employs advanced control systems, like an optimal linear quadratic regulator and a well-damped lead-lag controller, to ensure the nanorobot's precise positioning. The proposed system integrates a unique kind of particle swarm optimization, specifically the hybrid weighted quantum particle swarm optimization (WQPSO) with fuzzy logic to enhance the nanorobot motion control and optimize the route planning in uncertain conditions. By leveraging the principles of quantum computing, particle swarm optimization, and fuzzy logic, the aim of this paper is to boost the efficiency and adaptability of path planning algorithms. The results of the rigorous experiments which were carried out demonstrate the hybrid method's effectiveness in enabling nanoscale robots to navigate in environments with complex and randomly occurring obstacles. This paper shows that using the hybrid WQPSO and fuzzy logic can help control the movement of nanorobots accurately, which leads to an important progress in their ability to move autonomously at the nanoscale.

Keywords: Hybrid Weighted Quantum Particle Swarm Optimization, Fuzzy Logic, Nanoscale Robotics, Motion Control, Path Planning, Random Environmental Obstacles.

1. Introduction

Among the most important domains of research and development related to the use of nanotechnology is manufacturing at the nanoscale, which has profound effects on society and the economy. The recent improvements in Atomic Force Microscopy (AFM) and Scanning Tunneling Microscope (STM) technology have resulted in the development of several tools that can analyse and measure things at the atomic level (Amor et al., 2023). Furthermore, there are several significant drawbacks to the STM/AFM piezoelectric actuators' use of Lead Zirconate-Titanate (PZT) ceramic materials. It features a high voltage of operation, a small linear range, hysteresis, and thermal drift. These limitations hinder the employment of these types of SPM-based manipulators as standalone nano-manufacturing instruments. Therefore, to satisfy the stringent requirements of nanoscale manufacturing, unique actuation and sensing techniques are required.

Researchers have successfully applied the magnetic levitation technology for nanopositioning. Multiple research teams developed precision positioning instruments employing this approach. The Maglev stage of development, which was employed in this study is based on the developed model. Magnetic levitation's primary advantage over other existing technologies is that it operates in a noncontact manner, meaning that the required forces are supplied to the element that is in movement without the necessity of a mechanical contact. Consequently, there isn't any backlash, hysteresis, or friction. The Maglev technology doesn't produce wear particles or need lubricants; it can be used in clean-room conditions and for the operating voltage. Furthermore, the absence of intricate mechanical components significantly reduces the fabrication cost.

The Maglev nanopositioning device can be used in various ways, including for reducing vibrations for delicate instruments, scanning and imaging, manipulating atoms, and in techniques like micro-stereolithography (μSTL) and dip-pen lithography (DPN) and nanolithography. Thus, the Maglev stage must undergo extensive testing for set-point alterations, and motion planning and application-specific control algorithms must be developed to demonstrate its use as a tool in all its related applications.

https://doi.org/10.24846/v34i2y202506

The rest of this paper is organized as follows. Section 2 reviews related works on the advances in surveillance and robotics utilizing machine learning and optimisation techniques. Section 3 introduces the proposed system which combines fuzzy logic with WQPSO for nanorobot path planning in dynamic and uncertain environments.

In Section 4, the simulation results are discussed, which demonstrate the proposed system's higher performance in comparison with the traditional methods in terms of computation and path efficiency. Finally, Section 5 provides the conclusions of this paper and future research directions, highlighting potential applications in nanoscale robotics and other fields.

2. Related Works

Surveillance and mapping are key responsibilities in the majority of applications, including precision agriculture, remote sensing, tracking biodiversity, and military surveillance (Sun et al., 2023). Manual operations take too much time and are not viable in the majority of scenarios for accessing remote regions, hence small UAVs with a camera become the cost-effective solution (Petrescu et al., 2024). Despite that, UAVs suffer from heavy demands, i.e. fuel shortages, depending on ground vehicles or refueling stops in the case of long-range missions (Vellela et al., 2023; Wang, Z. et al., 2023). Figure 1 is an example of a UAV based coverage scenario with a UAV required to visit all intended targets (red squares) while depending on ground vehicles for recharging its batteries. This underscores the need for fuel-limited routing and multi-objective evolutionary algorithms to maximise UAV missions (Steinmann et al., 2023; Shi et al., 2023).

The recent developments in machine learning and computational intelligence have improved the functionality of UAVs and robots tremendously. For instance, colloidal robotics has been researched, in which context the ability of nanoscale systems in sophisticated environments has been emphasised (Liu et al., 2023). Smart methods have been optimised for applications related to underwater wireless sensor networks, with an emphasis on the universality of such methods in different applications (Zhang et al., 2023). Similarly, ANN-based control using an adaptive neuro-fuzzy inference system (ANFIS) is achieved through enhanced hybrid meta-heuristic tuning methods that increase control system performance during dynamic operations (Verma & Valluru, 2023). Such methods are part of the general trend toward the utilisation of computational intelligence towards robot system optimisation (Dulhare & Houssein, 2023).

During the manufacturing and production phase, innovations in lattice-materials have also been found when applied to light and efficient robot design (Almesmari et al., 2023). Computational intelligence has been responsible for modeling lightweight composite materials which form the core of UAV and robot performance enhancement (Amor et al., 2023). These approaches are accompanied by advancements in miniature soft robots, which feature a new potential for actuation, fabrication, and control (Chi et al., 2024). Distributed DQN methods have been used to implement smart cooperative robot control, demonstrating machine learning's

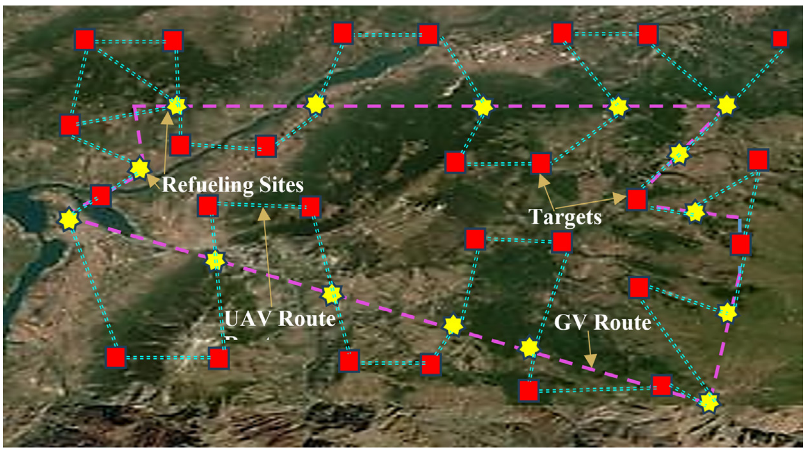


Figure 1. An example of a UAV-based coverage program

strength in decision-making optimisation (Wei & Cheng, 2025). Swarm robot intelligence in distributed manufacturing systems has also been researched, emphasizing the method's scalability in manufacturing (Petrescu et al., 2025).

The role of sensor integration and decision-making in real-time applications is equally important, aided by machine learning. Robotic morphing surfaces have been produced with machine learning assistance, which indicates the flexibility of intelligent systems in changing environments (Wang, J. et al., 2023). The application of image processing methods for IoT security has been discussed, which is essential for stable UAV operations (Al-Ghaili et al., 2023). Also, the integration of sensing components into iontronic skins for robotics has been introduced, which provides strong and sensitive feedback systems (Shi et al., 2023).

Path planning is still an essential challenge, and Voronoi diagrams have become a well-known method even despite their downsides, e.g. sharp turns and complicated implementation (John et al., 2025; Chi et al., 2024). The employed techniques are accompanied by innovations in miniature soft robots, which feature a new potential in actuation, fabrication, and control (Chi et al., 2024).

3. Proposed System

This research work combines fuzzy logic with the hybrid WQPSO (Weighted Quantum Particle Swarm Optimization) approach to improve how nanorobots plan their paths and move in unpredictable settings. The solution provides an answer to the navigational issues of nanorobots in uncertain and complex environments, particularly in medical, manufacturing, and environmental monitoring applications. The paper focuses on the development of an integrated method to improve robot mobility accuracy, and path planning optimisation for small robots, with a view to implementing the proposed approach under various conditions.

This approach assumes prior knowledge of static obstacles by the nanorobot by using image processing related to the workspace. The Dijkstra algorithm calculates the shortest path using the Harris corner detection method, which builds a visibility graph and adds known static barriers. The embedded nanosensors detect real-time dynamic barriers. Eight sensors are placed on each nanorobot, evenly covering the front and back half-spaces, as shown in Figure 1. The sensors provide critical information regarding the moving barriers such that the nanorobot can navigate dynamically. The Harris corner detection method is chosen because it can be implemented suitably under changing lighting conditions, rotation, or translation conditions.

Harris's technique is better than Moravec's corner detector as it includes corner score differences and direct directions for the purpose of robust corner detection. For finding local maxima when detecting candidate regions of interest, the Harris method is utilized and thresholding or clustering reduces the number of detected points in the analyzed set. This method can even be applied to 3D spaces, which makes it even more adaptable. By combining fuzzy decision-making and WQPSO optimisation, this method uses the strengths of both algorithms to address the navigation challenges faced by nanorobots. Based on the compatibility of these combined algorithms, the method promises to come up with a strong, efficient, and agile solution to path planning if the environment is dynamic and uncertain.

It is assumed that I is a 2D grayscale image:

$$i_u = \begin{cases} 1 & \rho \le S_x \\ 0 & otherwise \end{cases}$$

and that a picture patch is shifted by (x, y) over the area (u, v):

$$S(i,j) = \sum_u \sum_v w(u,v)\left(X(u+x, v+j) - X(u,v)\right)^2 \quad (1)$$

where w(u,v) is the squared difference based on the weight assigned to each pixel.

Let X_i and X_j be partial derivatives of X. Then $X(u+x, v+j)$ namely the Taylor expansion can be expressed as follows:

$$X(u+i, v+j) \approx X(u,v) + X_i(u,v)i + X_j(u,v)j \quad (2)$$

By replacing equation (2) with equation (1), the following is obtained:

$$S(i,j) \approx \sum_u \sum_v w(u,v)(X_i(u,v)i + X_j(u,v)j)^2 \quad (3)$$

The Harris matrix is defined based on the sensor A:

$$A = \sum_u \sum_u w(u,v)\begin{bmatrix} X_i^2 & X_i X_j \\ X_i X_j & X_j^2 \end{bmatrix} = \begin{bmatrix} X_i^2 & X_i X_j \\ X_i X_j & X_j^2 \end{bmatrix} \quad (4)$$

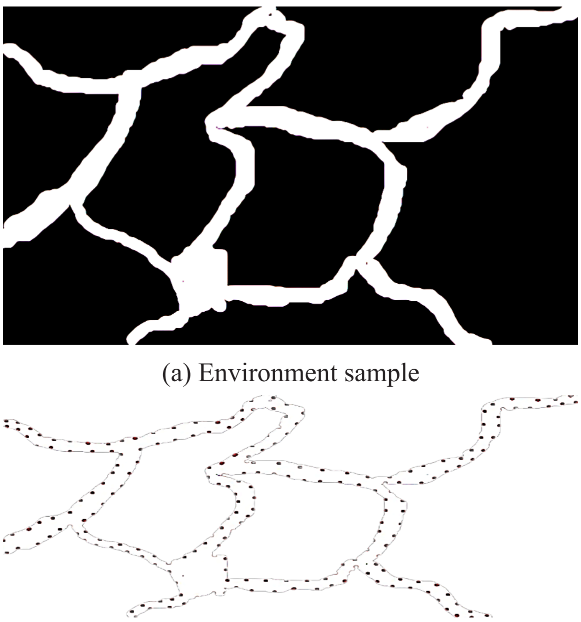
The matrix form in equation (4) is expressed as:

$$S(i,j) \approx (i,j) A \begin{pmatrix} i \\ j \end{pmatrix} \quad (5)$$

Since the computation of the necessary eigenvalues is computationally expensive, the following function M_c with a configurable sensitivity value λ is defined:

$$M_c = \lambda_1\lambda_2 - l(\lambda_1 + \lambda_2)^2 = \det(A) - l\,trace^2(A) \quad (6)$$

This function effectively determines the determinant and facilitates the tracing of A in identifying the interest points or corners, as it does not necessitate the computation of matrix A's eigenvalue decomposition. It was found that the values within the range of 0.04-0.15 are viable, although one has to determine the value experimentally. The environment corner, depicted in Figure 2(a), can be located using the Harris method, as illustrated in Figure 2(b). The proposed method increases the environment size effectively after making the visibility graph with the Harris method, as shown in Figure 3, to ensure a reliable path through the unchanging environment.



(a) Environment sample

(b) Harris corner detection

Figure 2. Detection technique (Harris corner detection)

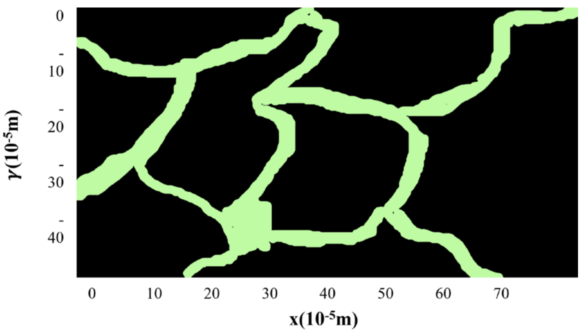


Figure 3. Visibility graph

3.1 Initial Path Planning with Dijkstra's Algorithm in the Presence of Static Obstacles

It is assumed, based on Figure 3, that the destination point's coordinates are (73, -15) and the starting point's position is (2, –37) for a given path planning in the given example setting. For the expanded sample environment, a visibility graph is generated once the corners of the area and its extension have been determined using the Harris method. Figure 4 displays the visibility graph for the expanded sample environment. It should be mentioned that Figure 4 shows the expanded surroundings without it, providing a more detailed view of the visibility graphs. It should be noted that Figures 3 and 4 show a cross-section of an artery generated by a blood vessel and a virtual plane intersecting with each other. The system applies the Dijkstra algorithm to identify the shortest path for the sample environment. The visibility graph for this simulation included 1009 edges and 154 nodes.

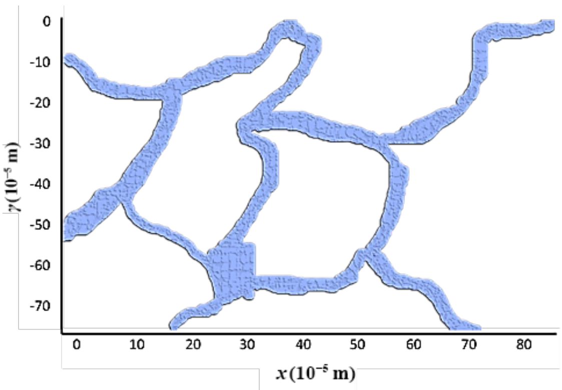


Figure 4. Colour border on the sample environment

3.2 Path Planning Model Which Was Used for the Hybrid WQPSO-Fuzzy Logic Approach

This subsection focuses on a simulation of randomly arranged particles in a given environment. Using P-models to display the positions of the particles and perimeter formulas, these particles generate random numbers through the cellular automata theory (CA theory). The changeover is independent of the impediments in each area and real-time topographic data for the entire surface. The architecture of the proposed model is shown in Figure 5.

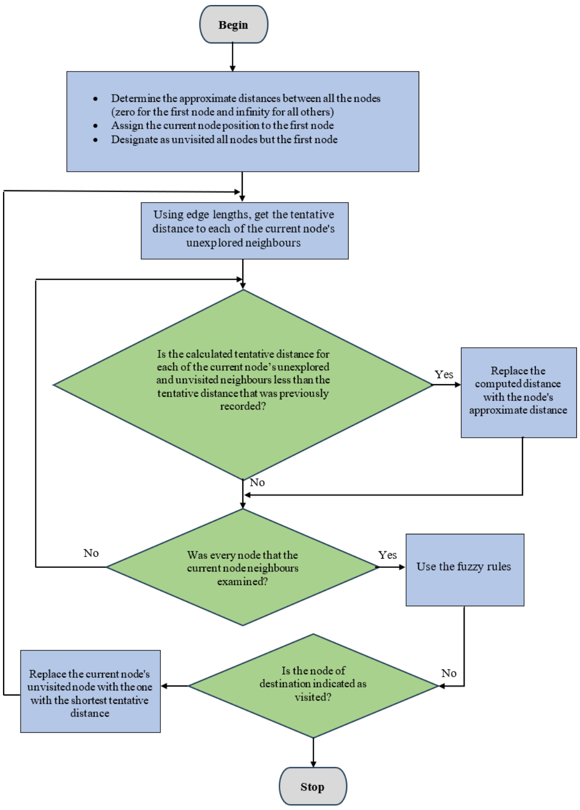


Figure 5. Flow Diagram of the proposed model based on a Hybrid WQPSO-Fuzzy Logic Approach for Dynamic Path Planning

The rules governing the automatons' states, which remain unchanged during this procedure, provide the stability of the manipulation process. The environment in this model can be represented as Figure 6 for a probability vector.

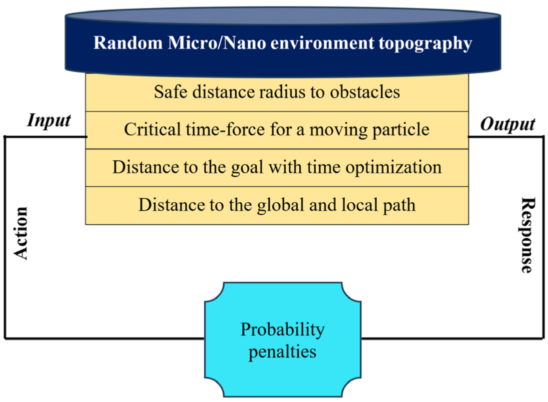


Figure 6. Fuzzy rules in nanotechnology for path planning

The median penalty number of automatons is M(n), which is expressed as follows:

$$M(n) = E(\beta(n)/p(n)) = Pr(\beta(n) = 1/p(n)) \quad (7)$$

$$\sum_{x=1}^{r} Pr\left(\beta(n) = \frac{1}{\alpha(n)} = \alpha_x\right) = \sum_{x=1}^{r} c_x p_x(n) \quad (8)$$

where $\beta = \{0,1\}$ represents outputs, $a = a_1,\ldots,a_r$ denotes actions set as Input and $c = \{c_1,\ldots,c_r\}$ indicates the probability penalty. This straightforward automation performs better than a probability-based automaton with an unoptimised behavior. The likelihood that a_x receives the penalty signal is c_x. The following equations describe the optimal automaton:

$$\lim_{n\to\infty} E(M(n)) = c_x \quad (9)$$

$$c_x = min\{c_x\} \quad (10)$$

The employed technique aims to identify the shortest path between the starting and ending points, to take into account the associated complexity of obstacles, determine the shortest path between the starting point and any random point, and address the problem of micro-/nanoparticle path planning based on the crucial time-force problem. Subsection 3.3 incorporates the key time-force limitation, crucial for navigating particles shaped like a sphere, into this model. The pathways are analysed through this automaton-based approach, and as soon as a path is not optimal because of crucial time limitations (resulting from dynamic manipulation), the search process is repeated until the best path for each path segment is found. One approach for resolving optimisation issues consists in utilizing the optimisation algorithm for population particles. The competency function is used to calculate the competency value of the whole particle; typically, their competency value is two. The velocity of each particle determines its direction.

The WQPSO–Fuzzy algorithm begins the process with a set of responses (called particles) and updates these particles with each iteration to produce optimal outcomes. If the decision functions and particle locations are identical, equations (11) to (14) will be utilised to determine the velocity and position vectors of the particles for each iteration:

$$V_u = w \cdot V_{u1} + c_1 \cdot r_1 \cdot (pBest_x i_u) + c_2 \cdot r_2 \cdot (nBest_x i_u) \quad (11)$$

$$V_{min} \le V_u \le V_{max} \quad (12)$$

$$S_x = \frac{1}{1} + e^{-V_u} \quad (13)$$

$$i_u = \begin{cases} 1 & \rho \le S_x \\ 0 & otherwise \end{cases} \quad (14)$$

According to equation (11), each particle's new velocity vector is defined by its previous velocity vector. The particle with the best position is termed as "$pBest_x$," and the particle that is closest to the main particle is termed as "$nBest_x i_u$." In the event that every particle's neighbour encompasses the entire group, the particle's optimal location inside the group of particles is termed as "*gBest,*" "$gBest_x$" being used to indicate the best position of a particle. Each particle's velocity vector is transformed into a random velocity function using equation (14).

The path planning method is constrained via the dynamic model's key time, and the path is optimized locally by taking into account the travel time between the starting point and the ending point, such as X0 and X1 represented in Figure 7.

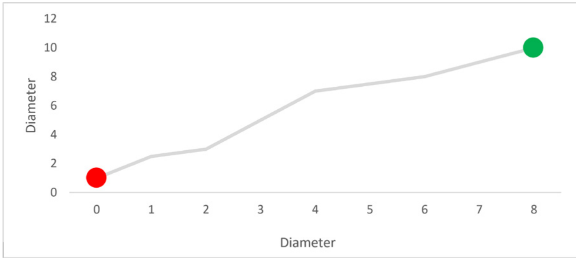


Figure 7. Path creation for both local and global searches (Red (source) and Green (destination))

The simulation is performed using the MATLAB platform, which is illustrated in Figure 8.

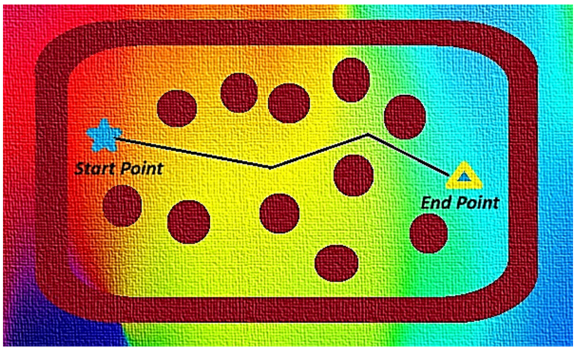


Figure 8. Virtual path-planning in a region of interest (ROI) exploiting surface barriers through the Hybrid WQPSO-Fuzzy Logic approach

Algorithm 1 outlies the steps for finding the shortest constrained time path to the target while avoiding obstacles:

Algorithm 1. Constrained Shortest Time to Target Algorithm for Two Elliptical Obstacles

Step 1: Initialize

Step 2: Input: Regions of ellipses; centre coordinates; axes (minor and major); orientation of angles; Starting points *S(x,y)*; Destination points *D(x,y)*

Step 3: Constraints checking

{

Step 3.1: if no region have constraint then go to step 4

}

else

{

Step 3.2: if single region have constraint then go to step 5

}

else

{

Step 3.3: if both regions have constraint then go to step 6

}

Step 4: Represent the environment as a grid or a continuous space, depending on the level of granularity required. For nanoscale robotics, a continuous representation might be more appropriate.

{

Step 4.1: Compute Euclidean distance for the path between *S* and *D*

}

Step 5: Define the motion constraints of the nanorobot, considering factors such as maximum speed, acceleration, and turning radius. For nanoscale robotics, need to consider factors like Brownian motion or other stochastic effects.

Step 6: Analyse the direct path region from *S* to *D* based on constraint.

{

Step 6.1: Identify the constraint region of free paths based on current obstacle.

Step 6.2: If any obstacles of two paths constrained then go to 6.3

Step 6.3: Compute the path length and coordinate points of path stored in the result

Step 6.4: Obstacle free subpaths obtained

Step 6.5: Link the subpaths into main path

Step 6.6: Compute the path length and coordinate points of path stored in the result

Step 6.7: Check if the 2nd main path has constraints because of obstacles (second) then repeat steps 6.4 to 6.6

Step 6.8: Read the path results file and arrange the paths in increasing order based on length of the path.

}

Step 7: Adapt the CSTT algorithm to generate a collision-free path from the starting point to the target point while avoiding the elliptical obstacles. This might involve discretizing the space around obstacles and finding a path that minimizes the distance travelled while adhering to the motion constraints.

Step 8: Once a collision-free path is found, optimize it for the shortest time to reach the target point. This could involve minimizing the time taken to traverse each segment of the path while considering the motion constraints.

Step 9: Two constraint regions numbered based on the distance from *S* to *D*.

{

Step 9.1: First region (nearest one)

Step 9.2: Second region (farthest one)

}

Step 10: Compute the path length and coordinate points of path stored in the result

Step 11: Find the alternate feasible obstacle-free paths from result file which are rendered in order

Step 12: End

3.3 Designing a fuzzy controller

This subsection is concerned with developing a 2D fuzzy controller to steer a nanorobot from an initial position to a target position securely. By incorporating two fuzzy controllers for vertical and horizontal path planning, the model can be extended from 2D to 3D, as shown in Figure 9. A controller will have to take care of movements like turning, acceleration, and braking based on parameters like speed, the distance to an obstacle, steering angle, direction, goal angle, and the distance to the goal, some of which are illustrated in Figure 10.

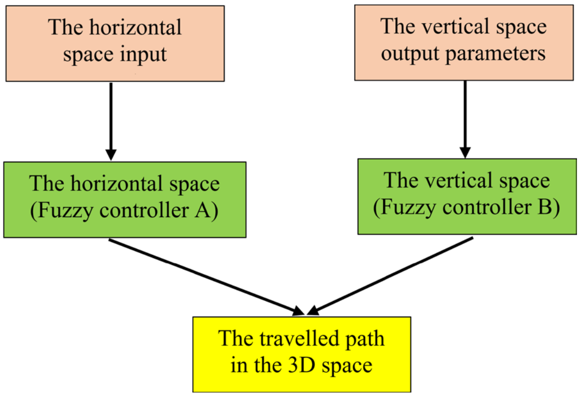


Figure 9. Integrating two fuzzy controllers to move along a path in a three-dimensional space

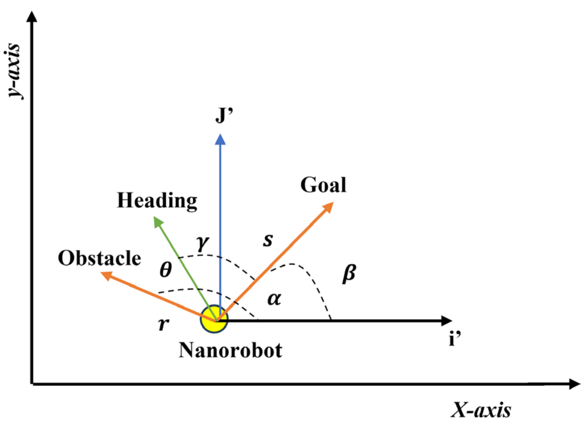


Figure 10. The parameters for the nanorobot's control

Modeling how robots behave by using analytical techniques is very complicated and requires a lot of computing power, while fuzzy logic, which mimics human decision-making with simple language rules, provides the right answers. These rules, written by specialists, utilise antecedents (fuzzy inputs) to deduce consequences (fuzzy outputs). Fuzzy logic is highly useful for decision-making in real time, especially under conditions of uncertainty or imprecision and is therefore ideally suited for intricate path-planning problems.

However, hindrances like environment data vagueness, rule description vagueness, and process vagueness in reasoning have to be eliminated so that the quality of the robot's navigation decisions be not compromised. The design procedure involves determining the fuzzy set of linguistic conditions, developing a rule-based model, and calculating the correlation between linguistic terms and precise quantities, with the aim of obtaining proportional output values based on the sensor inputs. In the second stage of the fuzzy logic process, as it can be seen in Figures 11 to 13, a membership feature for each phase has its m(x) determined.

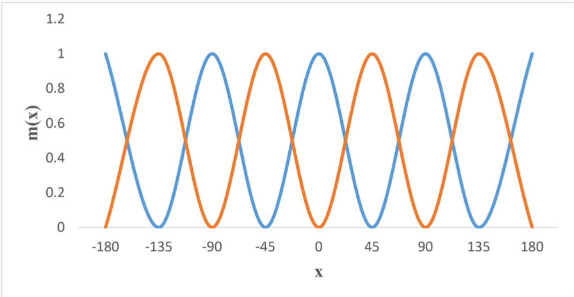


Figure 11. A nanorobot's angles related to the membership functions

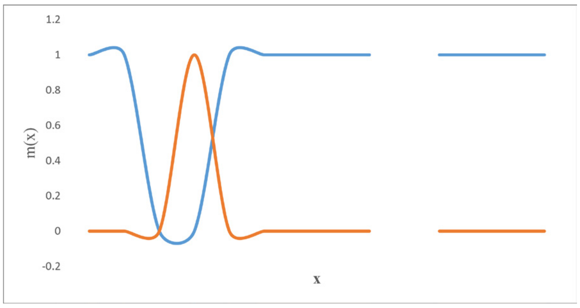


Figure 12. The distance between the nanorobot and the goal

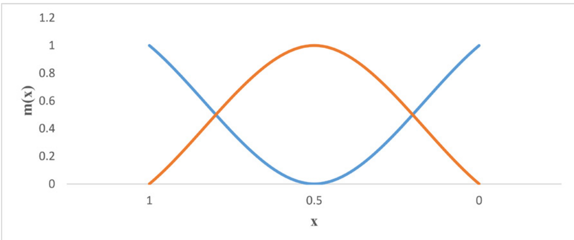


Figure 13. Nanorobot velocity and the identified obstacles

The envisioned technique is applicable to nanoscale robotics but it can also be utilized in larger-scale systems, thus providing a widely applicable method for path planning across a wide variety of applications.

4. Results and Discussion

4.1 Setup of the Simulation Process

The simulations were run on two separate computer systems. System 1 is a Windows 10 HP Spectre 360 laptop running with an Intel i7 7500U processor, which has two cores and a RAM capacity of 16 GB. It is suited for light computing tasks. For computationally demanding simulations, System 2 was utilised, a high-performance cluster node with 20 cores and a RAM capacity of 96 GB, running CentOS 6.5. Table 1 shows the instance solved to optimality within the range of 7200 seconds.

The pink-colored lines represent the network of roads, while the pink squares represent potential refueling sites. The dashed yellow lines represent the UAV route, the yellow triangles represent data points, and the red squares represent the refueling sites. Additionally, Figures 14(a) and 14(b) depict the candidate refueling sites, shown in pink, and the chosen refueling sites, displayed in red. The Mixed-Integer Linear Programming (MILP) solver found the UAV paths for the same situations, using the simulated values U = 20, and R = 15 for road network 1, and R = 10 for road network 2, shown in Figures 14(c) and 14(d), where t is the time necessary for detecting the new obstacle. This value indicates the undiscovered obstacle detection time, t indicates the previous obstacle time, λ indicates the nanorobot encounters the impediment in the interim, and r is a uniformly distributed random variable.

Table 1. The simulation process

n	Edge MILP		Node MILP	
	Road-nw 1	Road-nw 2	Road-nw 1	Road-nw 2
3	100	87.51(14.32)	100	87.51(14.32)
4	96.82	78.18(20)	99.89	81(20)
5	14.12	0(34.32)	36	0(34.32)
6	0	0(34.32)	0	0(34.32)

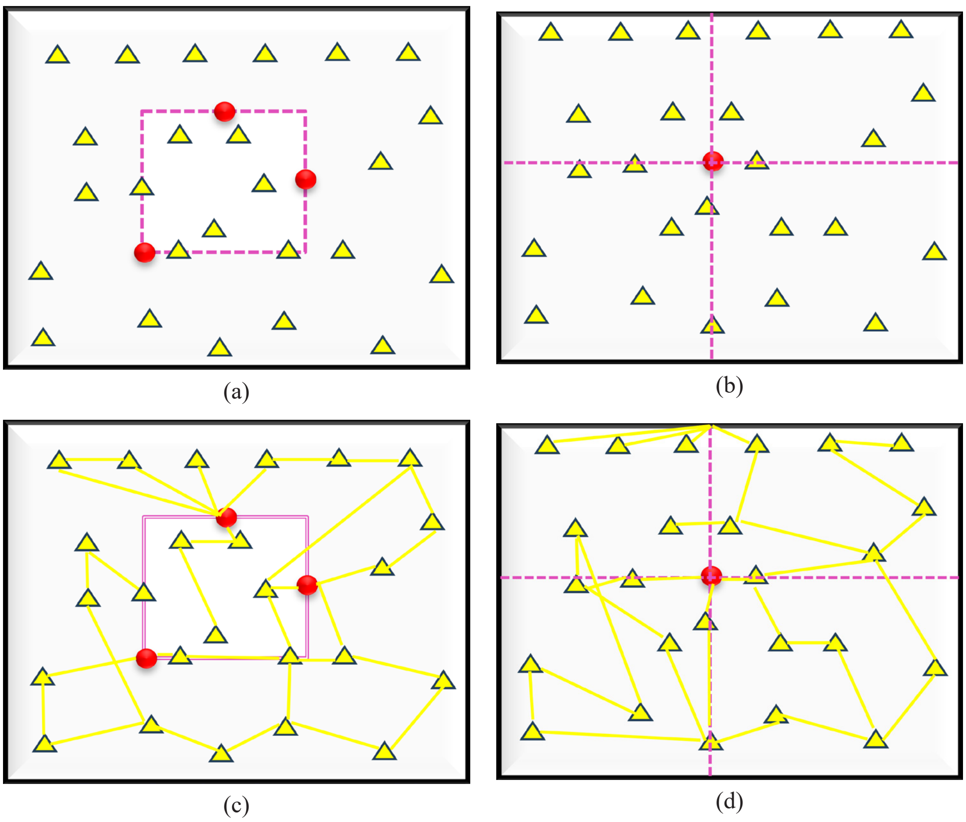


Figure 14. Sample cases utilised in the simulations

In Figure 15, the path of the nanorobot, using the five-stage WQPSO-fuzzy logic technique and two strategies, is shown by small black circles.

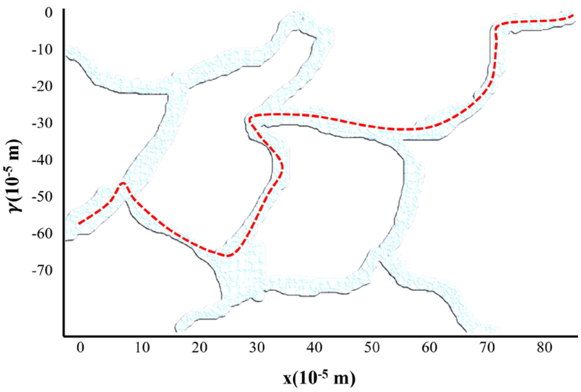


Figure 15. The nanorobot's path as it was determined by employing the proposed WQPSO-fuzzy logic approach

The ideal path derived by the Dijkstra algorithm is depicted in Figure 16 as the red path, which is made up of straight lines. The cross-section of Figure 16 displays the traversed path.

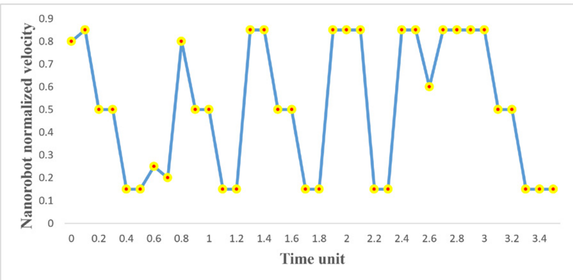


Figure 16. Normalized speed of the nanorobot on the traversed path

The proposed WQPSO-fuzzy logic multi-structure method consumes a lot of time and consistently requires longer computation times as it can be seen in Figure 17. In low-complexity situations, the PSO fails to produce better results as it can be seen in Figure 18 even though it takes longer to compute, as shown in Figure 17.

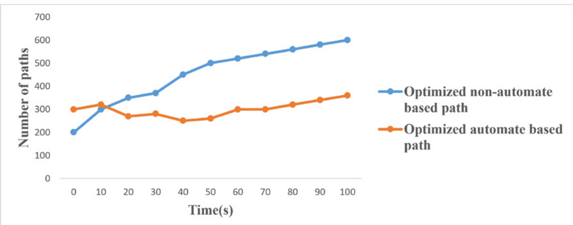


Figure 17. Simulations for the proposed WQPSO-fuzzy logic approach and scale algorithms' path segment count

On the other hand, as shown in Figure 18, the cellular automata algorithm makes use of an optimised path with a minimum number of segments, which results in a large reduction in manipulation time and an increase in precision. This result demonstrates that even with a higher computing time to start with, the proposed WQPSO in combination with a fuzzy logic technique can determine an optimised path and performs better in challenging tasks.

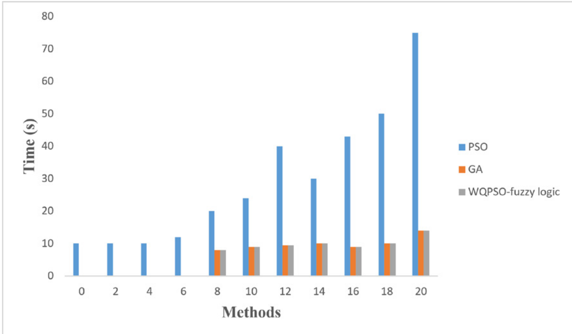


Figure 18. Computing times for the proposed WQPSO-fuzzy logic algorithm in comparison with PSO and GA

Figure 19 displays the nanorobot's heading angle during the simulation.

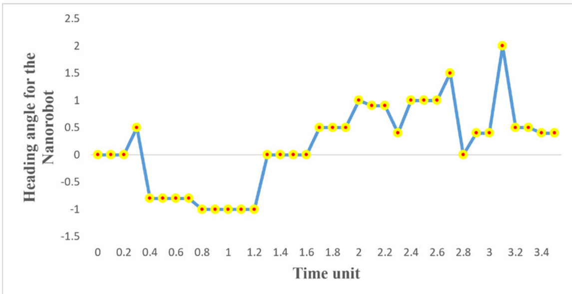


Figure 19. Heading angle of the nanorobot during the simulation

Figure 20 displays the changes in the local goal angle, highlighting the dynamic adjustment process during the operation.

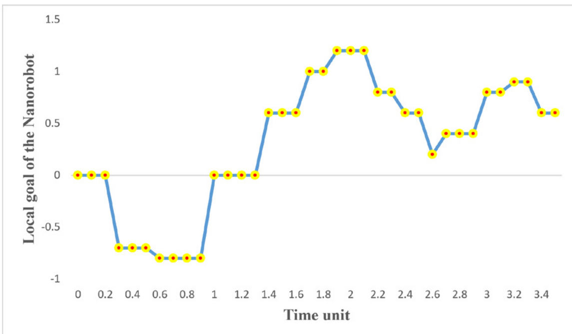


Figure 20. Local goal of the Nanorobot during the simulation

Figure 21 depicts the traveled path in the time domain for a clearer understanding.

In Environment 2, the proposed WQPSO-fuzzy logic approach was applied with the purpose of demonstrating its effectiveness. Figure 22 displays the obtained path. The length of the traveled path in this simulation is around 244.8309 units, while the length of the major offline path generated by the proposed method is approximately 242.297 units. The findings demonstrate the superior efficacy of the proposed approach over the previously employed methods, such as the Genetic Algorithm (GA), Multi-Operator-Based Simulated Annealing (MSA), and Simulated Annealing (SA) techniques. Table 2 presents the outcomes for the four above-mentioned approaches with regard to the offline processing time, online processing time and path length. A machine equipped with a 2.8 GHz Core 2 Duo CPU and 2 GB of RAM provided all these findings.

Further on, a quantitative comparison of the four algorithms with respect to computation time, path length, and energy consumption is illustrated in Table 3. From the results, it is clear that the proposed method achieves a reduction in the computation time by 40%, an increase in path efficiency by 6.5%, and a reduction in energy consumption by 20% in comparison with GA, SA, and MSA. These results are achieved because the proposed method is capable of optimising path planning with a lower number of iterations and because it employs lightweight fuzzy logic rules to deal with dynamic obstacle avoidance.

It can be seen that the proposed method is better than the traditional methods with regard to all the performance measures. Its computation time of 0.129 seconds is significantly lower than that obtained by GA (2.155 seconds) and SA (0.412 seconds), which is beneficial to real-time applications. Its path length of 246 units is lower than that obtained by GA (263 units) and SA (256 units), which indicates an enhanced path efficiency. The energy consumption of 50 J obtained by the proposed method is also

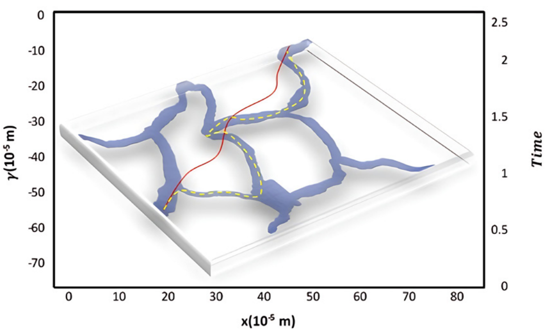


Figure 21. The travelled path in the time domain that was strolled while designating two distinct zones

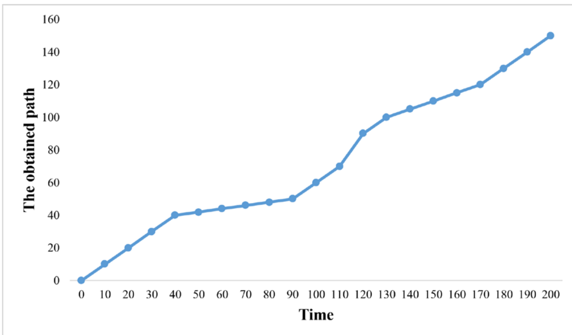


Figure 22. The path that was obtained in environment 2 by employing the proposed WQPSO- fuzzy logic method

Table 2. Comparison of the proposed path planning technique with the other three existing techniques

Parameter	WQPSO with fuzzy logic	MSA	SA	GA
Offline processing time (s)	0.0282	0.2283	0.3918	1.8026
Online processing time (s)	0.1291	0.2513	0.4116	2.1547
Path length (units)	245.841	247.25	257.80	262.53

Table 3. Comparison of Computation Time, Path Length and Energy Consumption for the proposed technique and the other three existing techniques

Method	Computation Time (s)	Path Length (units)	Energy Consumption (J)
WQPSO with Fuzzy Logic	0.129	246	50
GA	2.155	263	80
SA	0.412	256	70
MSA	0.251	247	60

lower than that obtained by GA (80 J) and SA (70 J), reflecting a higher energy efficiency. Furthermore, the WQPSO with fuzzy logic algorithm achieved a 30% reduction in memory usage, which highlights its suitability for real-time applications. Overall, these results demonstrate the efficacy and feasibility of the proposed technique for nanoscale robotics and other areas requiring an adaptive and efficient path planning.

5. Conclusions and Future Directions

The proposed WQPSO with fuzzy logic approach builds a path based on known static obstacles and uses nanosensors to identify dynamic obstacles, making nanorobots adaptable in uncertain environments. Local paths are derived from the global path, which reduces the overall cost related to the implementation of the algorithm, and optimises path efficiency and dynamic modeling, while also reducing the computational burden. The quantitative comparisons demonstrate the superiority of the proposed method, resulting in a 40% reduction in computation time, a 6.5% increase in path efficiency, and a 20% decrease in energy usage in comparison with conventional methods such as GA and SA. Its computational complexity, O(n * m + k), provides scalability for hardware-limited systems, and its lightweight nature reduces memory usage by 30%. Future research could focus on implementing this method to larger robotic systems and other applications such as autonomous vehicles, industrial robots, and robotic surgery. Field deployment challenges such as sensor limitations, environmental uncertainty, and power constraints will be addressed by using robust sensor fusion algorithms, adaptive control techniques, and energy-limited methods. The incorporation of real-time prediction of obstacles by machine learning and adaptive control will make this method even more applicable in dynamic environments. The further application of this method in fields such as environmental monitoring and precision agriculture will be carried out to enhance its applicability across a wide range of disciplines. The method tends to improve nanoscale robotics and enable productive, adaptive, and scalable path-planning techniques.

Acknowledgements

The authors acknowledge the Department of Electronics and Communication Engineering of Saveetha Engineering College in Chennai, Tamil Nadu, India, for their support and resources during the research that was carried out.

REFERENCES

Al-Ghaili, A. M., Gunasekaran, S. S., Jamil, N. et al. (2023) A Review on Role of Image Processing Techniques to Enhancing Security of IoT Applications. *IEEE Access*. 11, 101924-101948. https://doi.org/10.1109/ACCESS.2023.

Almesmari, A., Alagha, A. N., Naji, M. M. et al. (2023) Recent Advancements in Design Optimization of Lattice-Structured Materials. *Advanced Engineering Materials*. 25(17), 2201780. https://doi.org/10.1002/adem.202201780.

Amor, N., Noman, M. T., Petru, M. et al. (2023) A review on computational intelligence methods for modelling of light weight composite materials. *Applied Soft Computing*. 147(C), art. no. 110812. https://doi.org/10.1016/j.asoc.2023.110812.

Chi, Y., Zhao, Y., Hong, Y. et al. (2024) A Perspective on Miniature Soft Robotics: Actuation, Fabrication, Control, and Applications. *Advanced Intelligent Systems*. 6(2), art. no. 2300063. https://doi.org/10.1002/ais2.300063.

Dulhare, U. N. & Houssein, E. H. (eds.) (2023) Machine Learning and Metaheuristics: *Methods and Analysis*. Berlin, Springer Nature.

John, M. J., Dinakaran, K. & Bharathiraja, N. (2025) Advancing transport safety with faster pre-convoluted neural networks and lightweight multi-scale fusion for driver distraction detection. *International Journal of Heavy Vehicle Systems*. 32(1), 101-121. https://doi.org/10.1504/IJHVS.2025.144167.

Liu, A. T., Hempel, M., Yang, J. F. et al. (2023) Colloidal robotics. **Nature Materials**. 22(12), 1453-1462. https://doi.org/10.1038/s41563-023-01413-9.

Petrescu, M., Ștefan, M.-C., Panagoreț, A. A. et al. (2024) Route Planning and Machine Learning Algorithms for Sensor-Equipped Autonomous Vehicles in Agriculture. *Studies in Informatics and Control*. 33(4), 105-112. https://doi.org/10.24846/v33i4y202410.

Petrescu, A.-G., Oncioiu, I., Petrescu, M. et al. (2025) Integration of Robot Swarm Intelligence for Distributed Manufacturing Systems. *Studies in Informatics and Control*. 34(1), 37-46. https://doi.org/10.24846/v34i1y202503.

Shi, J., Dai, Y., Cheng, Y. et al. (2023) Embedment of sensing elements for robust, highly sensitive, and cross-talk–free iontronic skins for robotics applications. *Science Advances*. 9(9), art. no. eadf8831. https://doi.org/10.1126/sciadv.adf8831.

Steinmann, S. N., Wang, Q. & Seh, Z. W. (2023) How machine learning can accelerate electrocatalysis discovery and optimization. *Materials Horizons*. 10(2), 393-406. https://doi.org/10.1039/D2MH00813E.

Sun, H., Gao, X., Guo, L. Y. et al. (2023) Graphene-based dual-function acoustic transducers for machine learning-assisted human–robot interfaces. *InfoMat*. 5(2), art. No. e12385. https://doi.org/10.1002/inf2.12385.

Vellela, S. S., Sk, K. B., Venkateswara Reddy, B. et al. (2023) Introducing the Nano Cars Into the Robotics for the Realistic Movements. International Journal of *Progressive Research in Engineering Management and Science* (*IJPREMS*). 3, 235-240. https://doi.org/10.5281/zenodo.7898999.

Verma, N. & Valluru, S. K. (2023) ANN Based ANFIS controller Design Using Hybrid Meta-Heuristic Tuning Approach for Cart Inverted Pendulum System. *Multimedia Tools and Applications*. 83, 54839–54861. https://doi.org/10.1007/s11042-023-14003-4.

Wang, J., Sotzing, M., Lee, M. et al. (2023) Passively Addressed Robotic Morphing Surface (PARMS) Based on Machine Learning. *Science Advances*. 9(29), art. no. eadg8019. https://doi.org/10.1126/sciadv.adg8019.

Wang, Z., Dabaja, R., Chen, L. et al. (2023) Machine learning unifies flexibility and efficiency of spinodal structure generation for stochastic biomaterial design. *Scientific Reports*. 13(1), art. no. 5414. https://doi.org/10.1038/s41598-023-32269-1.

Zhang, Y., Liu, Z. & Bi, Y. (2023) Node deployment optimization of underwater wireless sensor networks using intelligent optimization algorithm and robot collaboration. *Scientific Reports*. 13(1), art. no. 15920. https://doi.org/10.1038/s41598-023-50999-9.