Bulletin of Electrical Engineering and Informatics
Vol. 15, No. 3, June 2026, pp. 2089~2135
ISSN: 2302-9285, DOI: 10.11591/eei.v15i3.11859

Adaptive bidirectional heuristic rapidly exploring random tree* for efficient path planning

Heru Suwoyo[1], Ahmad 'Athif Mohd Faudzi[2], Andi Adriansyah[1], Yudhi Gunardi[1], Julpri Andika[1], Yingzhong Tian[3]
[1]Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana, Jakarta, Indonesia
[2]Department of Control and Mechatronics Engineering, Faculty of Electrical Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia
[3]Department of Mechanical and Automation Engineering, School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, China

Article Info

Article history:

Received Nov 14, 2025
Revised Mar 14, 2026
Accepted Apr 19, 2026

Keywords:

Adaptive bidirectional heuristic-rapidly exploring random tree*
Adaptive rewiring
Bidirectional tree growth
Heuristic-based parent selection
Hybrid informed-fast sampling

ABSTRACT

Sampling-based path planning algorithms such as rapidly exploring random tree* (RRT*) are widely used for autonomous navigation in complex environments. However, many RRT variants suffer from slow initial exploration, suboptimal convergence, and search inefficiency in dense spaces. Based on this, adaptive bidirectional heuristic-RRT* (ABH-RRT*) is proposed. It is a novel method introduced as a unified path planner. ABH-RRT* integrates bidirectional tree growth, heuristic-based parent selection, fast-informed hybrid sampling, and adaptive reordering to improve exploration efficiency and path optimality. The algorithm speeds up the initial path recovery caused by the presence of dual tree expansion and fast sampling. In addition, the algorithm also refines the solution using informed sampling and adaptive reordering to improve convergence toward near-optimal paths. The performance of ABH-RRT* is evaluated in four environments with different complexity levels and compared with RRT, RRT*, Fast-RRT*, Smart-RRT*, and Informed-RRT*. Experimental results show that ABH-RRT* consistently produces shorter paths and faster convergence, reduces path cost by 2–24% and increases convergence speed by 40–58% in dense and constrained environments. These results show that ABH-RRT* is a better and adaptive solution for path planning in complex scenarios.

Corresponding Author:

Heru Suwoyo
Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana
St. Raya Meruya Selatan No. 1, West Jakarta, Special Capital Region of Jakarta 11650, Indonesia
Email: heru.suwoyo@mercubuana.ac.id

1. INTRODUCTION

Path planning is a key part of autonomous navigation systems. It plays an important role for things like self-driving cars, mobile robots, and unmanned aerial vehicles (UAVs) [1]–[6]. Essentially, the path planning is a task used to produce collision-free and nearly optimal path from start to goal point [7]–[9]. In practical scenarios, environment often has cluttered obstacles, narrow passages, and strange shapes. It makes the algorithm hard to explore and find the best path.

Many people now use sampling-based algorithms for motion planning because they can quickly search high-dimensional configuration spaces without having to fully discretize them. The rapidly exploring random tree* (RRT*) algorithm is one of these methods that has been used a lot because it can quickly expand the search tree into areas of the state space that haven't been explored yet [10], [11]. The classical

RRT algorithm frequently yields suboptimal paths, as it emphasizes exploration while lacking mechanisms for path refinement or optimization.

To overcome this limitation, various enhancements of RRT have been suggested. RRT* adds a rewiring mechanism that makes revision to the node connection which guarantees asymptotic optimality [12], [13]. By growing two trees at the same time from the start and goal configurations, bidirectional methods like RRT-Connect speed up the process of finding the first path [14]. While algorithms such as Smart-RRT* guide tree expansion using heuristic information to accelerate convergence toward promising regions [15]–[17]. Other approaches, including Informed-RRT*, restrict the sampling region to an ellipsoidal subset after an initial solution is found, thereby improving convergence speed during later iterations [18]–[20]. Additional variants such as Fast-RRT* focus on accelerating exploration [21]–[23], while hybrid approaches combine RRT with metaheuristic optimization techniques to improve path quality [24]–[26]. Anytime RRT* refines the path incrementally so that an initial solution can be obtained quickly, but the early path remains coarse [27]. LQR-RRT* uses a linear quadratic regulator [28] to improve trajectories. This method requires more computation that doesn't help with early exploration. Deterministic RRT and path-biased RRT, on the other hand, reduce randomness by using either deterministic or path-biased directionality. However, exploration flexibility goes down when the bias is wrong [29]. Grid-based/Lattice RRT and cost map RRT use grids or cost maps to help trees grow in the right direction. But because they depend on grid resolution, they are less flexible [30]. Sparse RRT limits the number of nodes to make calculations faster, but this could mean missing possible paths in areas with few nodes. It has also been suggested that combining metaheuristic and evolutionary algorithms could improve the quality of paths. Genetic algorithm-RRT* (GA-RRT*) uses genetic combinations to improve suboptimal paths [30]–[33]; particle swarm optimization-RRT* (PSO-RRT*) adjusts exploration bias using swarm intelligence [34]–[36]; and RRT applies pheromone mechanisms to strengthen exploitation in dense regions [4], [31], [37]. Other stochastic approaches such as Firefly-RRT [38], Cuckoo-RRT [39], and grey wolf optimizer-RRT assist the algorithm in escaping local minima and producing trajectories [40]–[43]. Additional acceleration variants, including Quick RRT*[44], more quickly RRT*[45], fast and flexible-RRT* (FF-RRT*) [46], and genetic algorithm optimization-RRT* (GAO-RRT*) [47], focus tree growth along the most promising paths using heuristic or global optimization techniques, thereby improving speed and path quality. Nevertheless, all these variants still face challenges such as slow initial exploration, suboptimal convergence to global solutions, static rewiring radius, and high computational complexity in cluttered or high-dimensional environments. Despite significant developments in RRT, several limitations remain. Generally, these approaches are limited to the use of a single approach, such as sampling efficiency, heuristic guidance, or bidirectional tree expansion strategies. Consequently, improvements at one stage of the planning process are less effective in terms of optimality and convergence compared to the initial conditions. Furthermore, the use of fixed parameters for expansion step size also limits the adaptability to complex environmental conditions.

To address these challenges, the adaptive bidirectional heuristic-RRT* (ABH-RRT*) is proposed. It is a unified sampling-based path planning framework that integrates bidirectional tree growth, heuristic-based parent selection, hybrid informed–fast sampling, and adaptive rewiring radius adjustment. The proposed method aims to improve both exploration efficiency and path optimality. These improvements are made by combining multiple complementary strategies within a single algorithmic framework. Bidirectional tree expansion accelerates initial path discovery and heuristic parent selection improves local cost optimization. Moreover, the hybrid sampling balances exploration and exploitation, and adaptive rewiring dynamically adjusts the search structure during tree expansion. The main contribution of this paper is summarized as follows:

- A unified ABH-RRT* framework that integrates bidirectional tree growth, heuristic parent selection, hybrid informed–fast sampling, and adaptive rewiring to improve motion planning efficiency.
- An adaptive parent selection and dynamic rewiring mechanism. These approaches aim to accelerate convergence while maintaining the asymptotic optimality property of RRT*.
- A hybrid sampling strategy combines quick global exploration with informed sampling. This approach aims to improve search efficiency after an initial path has been found.
- The algorithm's performance was evaluated through extensive experiments in four different environments. These environments varied in obstacle density and structural complexity. The results were then compared to those of RRT, RRT*, Fast-RRT*, Smart-RRT*, and Informed-RRT*.

Experimental evaluations were conducted in four environments with different obstacle densities. This evaluation is also structural complexities to assess the effectiveness of the proposed method. The results demonstrate that ABH-RRT* consistently improves path planning performance compared with conventional algorithms including RRT, RRT*, Fast-RRT*, Smart-RRT*, and Informed-RRT*. In dense and constrained environments, the proposed method reduces path cost by 2–24%. The algorithm improves convergence speed by 40–58%. In the most challenging environment containing narrow corridors, ABH-RRT* reduces the path

cost from 342.52 (Fast-RRT*) to 317.31. At the same result it is also requiring fewer iterations to obtain a feasible path. These results indicate that the proposed framework provides a more efficient and robust solution for autonomous path planning in complex environments.

2. METHOD

This section describes the method of the proposed ABH-RRT*. The proposed framework integrates bidirectional tree growth, heuristic parent selection, hybrid sampling strategy, and adaptive rewiring to improve both initial exploration and path convergence. The following subsections first review several representative RRT variants and then describe the proposed algorithm in detail.

2.1. Problem statement

The path planning problem can be formulated as follows. Given a state space $X \in \mathbb{R}^n$ with $n \in \mathbb{N}$ denoting the dimensionality of the space, it can be defined:

$$X = X_{\text{obs}} \cup X_{\text{free}} \tag{1}$$

where $X_{\text{obs}} \subset X$ represents the obstacle space and $X_{\text{free}} \subset X$denotes the collision-free space. Given a start point $x_{\text{init}} \in X_{\text{free}}$ and a goal point $x_{\text{goal}} \in X_{\text{free}}$, the path planning algorithm must find a continuous trajectory:

$$\sigma: [0, T] \to X_{\text{free}}, \tag{2}$$

such that $\sigma(0) = x_{\text{init}}$ and $\sigma(T) = x_{\text{goal}}$. The region surrounding the goal point x_{goal} is defined as the set:

$$X_{\text{goal}} = \{ x \in X \mid \| x - x_{\text{goal}} \| < r \} \tag{3}$$

where r represents the radius around the goal point that is considered to have reached the target area. Path planning aims to efficiently find a feasible trajectory from start to goal while ensuring completeness and optimality.

2.2. Baseline sampling-based algorithm

Sampling-based planners explore the configuration space by incrementally expanding a search tree in the collision-free space X_{free}. These methods avoid explicit discretization of the configuration space and are therefore well suited for high-dimensional motion planning problems. The RRT* algorithm builds a search tree, starting from the initial state x_{init}. Each iteration involves randomly selecting a state x_{rand} from the free space X_{free}. Subsequently, the nearest node x_{near} within the tree is identified, and a new node x_{new}, is generated by advancing from x_{near} toward x_{rand}, utilizing a predetermined step size η. Provided that the edge formed is devoid of collisions, the new node x_{new} is incorporated into the tree T. RRT demonstrates RRT demonstrates efficacy in exploring expansive configuration spaces and exhibits probabilistic completeness. Therefore, as the number of iterations increases, the chance of finding a working path from x_{init} to x_{goal} approaches one. However, the algorithm does not include mechanisms for path optimization, and therefore the resulting trajectory is typically suboptimal. The procedure of RRT is summarized in Algorithm 1.

Algorithm 1. RRT

```
Input: x_init, x_goal, N (maximum iteration), η (step size)
Output: Tree (T) from x_init
T ← InitializeTree(x_init)
for (i = 1) to (N) do
    x_rand ← uniformSampling(X_free)
    x_near ←neares(T, x_rand)
    x_new ← steer(x_near, x_rand, η)
    if collisionFree(x_near, x_new) then
        T ← addNode(T, x_new, x_near)
 end if
end for
return (T)
```

To enhance path optimality, RRT* incorporates a rewiring mechanism that iteratively improves the tree structure. Upon the generation of a novel node, denoted as x_{new}, the algorithm assesses the neighboring

nodes, X_{near}, situated within a specified radius, to identify the parent node that yields the minimal cumulative path cost. Should a connection with reduced cost be identified, the neighboring nodes are subsequently rewired via x_{new}. Consequently, this iterative process facilitates the enhancement of solution quality, and the algorithm's asymptotic optimality is assured with an increasing sample size. The procedure of RRT* is summarized in Algorithm 2.

Algorithm 2. RRT*

Input: x_{init}, x_{goal}, N (maximum iteration), η (step size)
Output: T_{opt}
$T \leftarrow$ InitializeTree(x_{init})
for $(i = 1)$ to (N) **do**
  $x_{rand} \leftarrow$ uniformSampling(X_{free})
  $x_{near} \leftarrow$ nearest(T, x_{rand})
  $x_{new} \leftarrow$ steer$(x_{near}, x_{rand}, \eta)$
  if collisionFree(x_{near}, x_{new}) **then**
    $X_{near} \leftarrow$ near(T, x_{new}, x_{near})
    $x_{min} \leftarrow \text{argmin}_{x \in X_{near}}\{c_{come}(x) + cost(x, x_{new})\}$
    Add x_{new} to T with x_{min} as parent
    if $x_{new} \in X_{goal}$
      update T_{opt}
    end if
  end if
end for
return (T)

Several extensions of RRT-based planners have been proposed to improve exploration efficiency and convergence performance. Fast-RRT accelerates the search process by applying focused sampling and path fusion mechanisms. Focused sampling biases the sampling process toward promising regions of the configuration space, allowing faster tree expansion. After a new node is added, the path fusion step combines candidate trajectories to generate a smoother and shorter path. The procedure of Fast-RRT is summarized in Algorithm 3.

Algorithm 3. Fast-RRT

Input: x_{init}, x_{goal}, N (maximum iteration), η (step size)
Output: T_{opt}
$T \leftarrow$ InitializeTree(x_{init})
for $(i = 1)$ to (N) **do**
  $x_{rand} \leftarrow$ fastSampling(X_{free})
  $x_{near} \leftarrow$ neares(T, x_{rand})
  $x_{new} \leftarrow$ steer$(x_{near}, x_{rand}, \eta)$
  if collisionFree(x_{near}, x_{new}) **then**
    $X_{near} \leftarrow$ near(T, x_{new}, x_{near})
    $x_{min} \leftarrow \text{argmin}_{x \in X_{near}}\{c_{come}(x) + cost(x, x_{new})\}$
    Add x_{new} to T with x_{min} as parent, named T_{new}
    $T \leftarrow$ pathFusion(T, T_{new})
    $T_{opt} \leftarrow$ optimizedPath(T)
  end if
end for
return (T_{opt})

Smart-RRT* improves convergence by incorporating beacon-guided sampling. Following the identification of an initial viable path, the nodes constituting the trajectory are designated as beacon nodes, thereby directing the sampling process toward regions of potential interest. Throughout each iteration, the sampling process alternates between beacon-guided sampling and uniform sampling, thereby establishing a balance between exploration and exploitation. Furthermore, the algorithm incorporates adaptive parent selection and rewiring to refine the tree structure. The Smart-RRT* procedure is concisely outlined in Algorithm 4.

Algorithm 4. Smart-RRT*

Input: x_{init}, x_{goal}, N (maximum iteration), η (step size)
Output: T
$T \leftarrow$ InitializeTree(x_{init})
$x_{beacon} \leftarrow \emptyset$
for $(i = 1)$ to (N) **do**

```
      if i=n+b, n+2b, n+3b, ... then
       x_rand ← BeaconGuidedSampling(X_free, x_beacon)
    Else
      x_rand ← UniformSampling(X_free)
    end if
    x_near ← nearest(T, x_rand)
    x_new ← steer(x_near, x_rand, η)
    if collisionFree(x_near, x_new) then
     X_near ← near(T, x_new, x_near)
     x_min ← argmin_{x∈X_near}{c_come(x) + cost(x, x_new)}
      T ←addNode(x_new, x_min)
      T ← rewiring(T, X_near, x_new)
      if initialPathFound then
        n ← i
    end if
    (T, directcost) ←pathOptimization(T, x_init, x_goal)
    if (directcost_new < directcost_old) then
      x_beacon ← pathOptimization(T, x_init, x_goal)
    end if
  end for
  return (T)
```

In contrast, Informed-RRT* improves convergence efficiency by restricting the sampling region once a feasible solution is found. After a path with cost c_{best} is obtained, the sampling process is limited to an ellipsoidal informed subset defined by the start and goal configurations. This subset contains all states that could potentially produce a path shorter than the current best solution. Sampling within this region allows the algorithm to focus the search on promising areas while maintaining the rewiring mechanism of RRT*. The procedure of Informed-RRT* is summarized in Algorithm 5.

Algorithm 5. Informed-RRT*

```
  Input: x_init, x_goal, N (maximum iteration), η (step size)
  Output: T
  T ← InitializeTree(x_init)
  InitialPathFound = 0
  for (i = 1) to (N) do
     if InitialPathFound then
       x_rand ← SampleInformedSubset(X_free, x_beacon)
     else
       x_rand ← UniformSampling(X_free)
    end if
    x_near ← nearest(T, x_rand)
    x_new ← steer(x_near, x_rand, η)
    if collisionFree(x_near, x_new) then
      X_near ← near(T, x_new, x_near)
      x_min ← argmin_{x∈X_near}{c_come(x) + cost(x, x_new)}
       T ←addNode(x_new, x_min)
       T ← rewiring(T, X_near, x_new)
       if x_new ∈ X_goal then
         InitialPathFound = 1
         T ←updatePath(T, x_new)
       end if
    end if
  end for
  return (T)
```

Although these variants improve specific aspects of sampling-based motion planning, they typically focus on a single mechanism such as exploration acceleration, heuristic sampling guidance, or path optimization. Consequently, their performance may still be limited in complex environments where efficient exploration and path optimization must be achieved simultaneously. This limitation motivates the development of a unified framework that integrates multiple complementary strategies to improve both exploration efficiency and path optimality.

2.3. Adaptive bidirectional heuristic-rapidly exploring random tree* (proposed method)

The proposed ABH-RRT* extends the conventional RRT* framework by integrating several mechanisms to improve exploration efficiency and path optimization. The algorithm integrates bidirectional tree growth, adaptive rewiring mechanism, heuristic sampling strategy. These components work together to

accelerate the discovery of an initial feasible path while maintaining efficient convergence toward near-optimal solutions in complex environments.

2.3.1. Bidirectional tree growth

In ABH-RRT*, two trees grow simultaneously from the start x_{init} and goal x_{goal} configurations. A random sample x_{rand} is drawn from the free space X_{free}, and the nearest node x_{near} in the active tree is found to generate a new node x_{new} using the steering function. If the new edge is collision-free, it is added to the tree, and a connection attempt is made to the opposite tree. In which the nearest node in active trees is measured by $x_{\text{near}} = \arg\min_{x \in T_{\text{active}}} \| x - x_{\text{rand}} \|_2$, followed by a steering step toward the new node $x_{\text{new}} = x_{\text{near}} + \eta \frac{x_{\text{rand}} - x_{\text{near}}}{\|x_{\text{rand}} - x_{\text{near}}\|_2}$. This dual-tree expansion increases the likelihood of rapidly finding an initial feasible path, especially in narrow corridors or environments with complex obstacles. This strategy can be illustrated in Algorithm 6.

Algorithm 6. Bidirectional tree growth

```
T_start ← InitializeTree(x_init), T_goal ← InitializeTree(x_goal)
for each x_i ∈ X_near do
  select T_active alternatively from T_start or T_goal
  x_rand ← FastSampling(X_free)
  x_near ← Nearest(T_active, x_rand)
  x_new ← Steering(x_near, x_rand)
  if collisionFree(x_new, x_near) then
    add x_new to T_active
    connect x_new to opposite T_active
  end if
end for
```

2.3.2. Adaptive parent selection

Each new node x_{new} is evaluated against a set of neighboring nodes X_{near} within a radius r. The algorithm selects the parent that minimizes the cumulative *cost-to-come* c_{come}:

$$x_{\text{min}} = \arg\min_{x \in X_{\text{near}}} (c_{\text{come}}(x) + \| x - x_{\text{new}} \|_2)$$

If a shorter path is found through x_{new}, the neighboring nodes are rewired accordingly:

if $c_{\text{come}}(x_{\text{new}}) + \| x_{\text{new}} - x_i \| < c_{\text{come}}(x_i)$, then rewire x_i to x_{new},

This adaptive mechanism preserves the asymptotic optimality property of RRT* while accelerating the convergence toward the optimal path. The adaptive parent selection procedure checks the new node x_{new} against its neighboring nodes within a certain radius to find the best parent node that lowers the total cost-to-come. The node x_{min}, which has the lowest total cost, is chosen to be the parent of x_{new}. Then, if connecting through x_{new} gives any neighboring node x_i a lower path cost, the algorithm does a local rewiring operation and updates the edge. This process makes sure that the tree structure gets closer to the best path configuration while still being asymptoticly optimal. Algorithm 7 illustrates how the process of adaptive parent selection works.

Algorithm 7. Adaptive parent selection

```
X_near ← Near(T, x_new)
x_min ← arg min_{x∈X_near} (c_come(x) + dist(x, x_new))
add x_new to T with x_min as parent
for each x_i ∈ X_near do
  if c_come(x_new) + dist(x_new, x_i) < c_come(x_i) then
    T ← rewire(x_i, x_new)
  end if
end for
```

2.3.3. Hybrid sampling strategy

The hybrid sampling strategy combines fast exploration before a feasible path is found with informed sampling after one is obtained. The informed subset is defined as:

$$X_{\text{informed}} = \{x \in X_{\text{free}} : c(x_{\text{init}} \to x \to x_{\text{goal}}) < c_{\text{best}}\},$$

where c_{best} denotes the current best path cost. This method keeps a balance between exploration and exploitation by first spreading the tree widely and then focusing on sampling within the ellipsoidal area that can improve the current best solution. Algorithm 8 gives more information about this process.

Algorithm 8. Hybrid sampling strategy

```
if T_opt exist then
  x_rand ← SampleInformedSubset(T_opt, x_goal, x_start)
else
  x_rand ← FastSampling(X_free)
end if
```

Algorithm 8 illustrates how the algorithm dynamically switches sampling strategies. Before a feasible path is found, it performs $FastSampling(.)$ over the entire free space X_{free}. Once an optimal path T_{opt} exists, the sampling is restricted to the informed subset around the current best path.

2.3.4. The adaptive bidirectional heuristic-rapidly exploring random tree* algorithm

The full ABH-RRT* framework sequentially applies the three integrated strategies. At each iteration, the algorithm alternates between two growing trees, selects random samples according to the hybrid strategy, and expands the active tree using adaptive parent selection and rewiring. Once a new node connects both trees, the optimal path tree T_{opt} is updated. Algorithm 9 depicts the ABH-RRT* working principle, where each iteration coordinates three main operations: expanding bidirectional trees, adaptively rewiring for cost efficiency, and using hybrid sampling that adjusts according to path availability. The overall workflow is summarized in Figure 1.

Algorithm 9. ABH-RRT*

```
T_start ← InitializeTree(x_init), T_goal ← InitializeTree(x_goal)
for each x_i ∈ X_near do
  select T_active alternatively from T_start or T_goal
  x_rand ← HybridSampling(T_opt, X_free)
  x_near ← Nearest(T_active, x_rand)
  x_new ← Steering(x_near, x_rand)
  if collisionFree(x_new, x_near) then
    add x_new to T_active
    connect x_new to opposite T_active
    if connected to opposite tree then
      update T_opt
    end if
  end if
end for
```

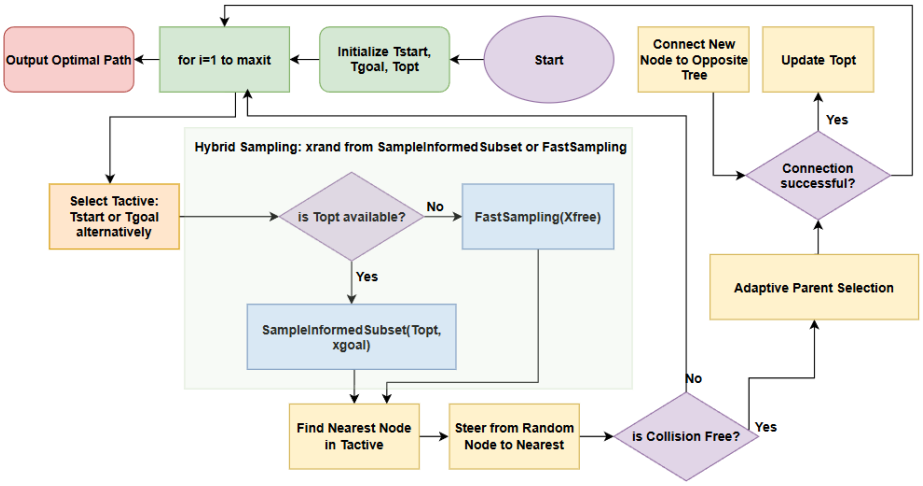

Figure 1. Overall workflow of ABH-RRT* algorithm, illustrating dual-tree expansion, adaptive rewiring, and hybrid sampling coordination

3. RESULTS AND DISCUSSION

3.1. Experiment setup

The efficacy of the proposed methodology was assessed across four simulated environments, each characterized by varying degrees of complexity. These environments were specifically constructed to emulate diverse navigation difficulties, encompassing open areas, moderately obstructed maps, densely populated obstacle fields, and confined corridor configurations.

All experimental procedures were executed within the MATLAB R2024b environment on a workstation featuring an Intel Core i7-14700K processor (3.40 GHz), 64 GB of RAM, and an NVIDIA RTX 3080 GPU. To facilitate a fair evaluation, all algorithms were subjected to identical parameter configurations, which included a step size of 4, a goal tolerance radius of 10, and a rewiring radius of 10. The performance of the proposed ABH-RRT* algorithm was evaluated in four simulated environments with increasing levels of complexity. The algorithms evaluated in this study include RRT, RRT*, Fast-RRT*, Smart-RRT*, Informed-RRT*, and the proposed ABH-RRT*. Performance was assessed using four evaluation metrics namely path cost, iteration to initial path found, computation time, success capability.

3.2. Environment 1: small square map

Environment 1 represents a relatively simple navigation scenario. It consists of a small square map with sparse obstacles. This environment primarily evaluates the ability of each algorithm to efficiently discover and optimize a path in an open configuration space.

Figure 2 illustrates the path planning results produced by six algorithms in this environment. Figure 2(a) shows the path generated by RRT, which quickly explores the free space but produces a relatively longer trajectory due to the absence of path refinement. Figure 2(b) shows the result of RRT*, where the rewiring mechanism improves path quality compared with RRT. Figure 2(c) presents the Smart-RRT* solution, where beacon-guided sampling accelerates convergence toward promising regions. Figure 2(d) shows Fast-RRT*, which emphasizes faster exploration. Figure 2(e) shows the path generated by Informed-RRT*, which focuses sampling on the current best solution. Finally, Figure 2(f) presents the trajectory generated by the proposed ABH-RRT* algorithm.

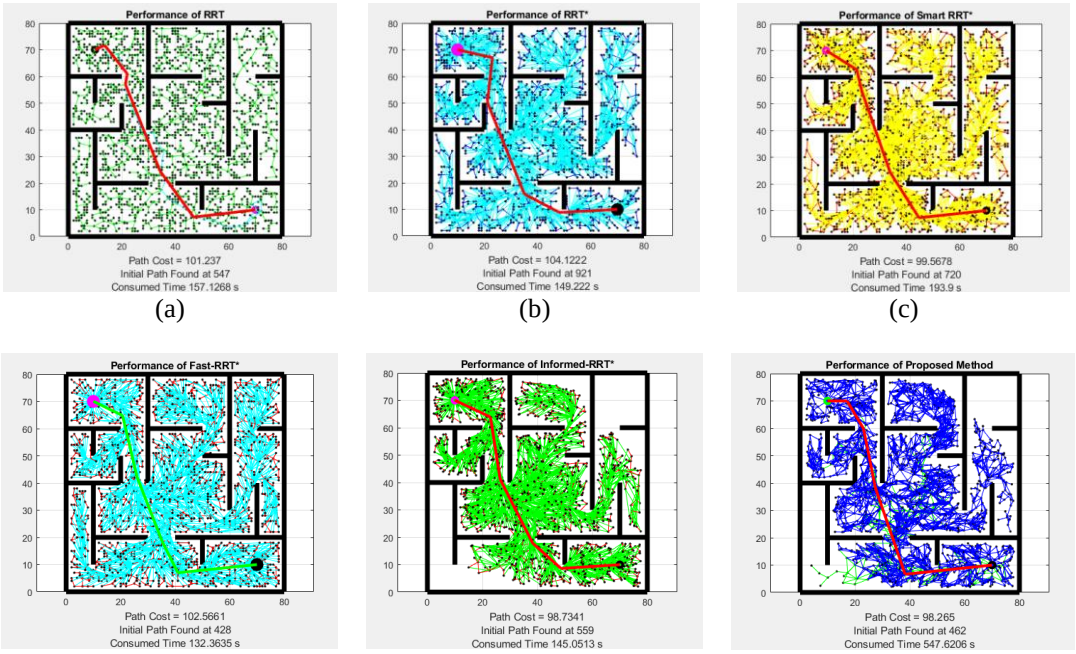

Figure 2. Path planning results of six algorithms in environment 1; (a) RRT, (b) RRT*, (c) Smart-RRT*, (d) Fast-RRT*, (e) Informed-RRT*, and (f) proposed method

The quantitative comparison in Table 1 indicates that ABH-RRT* achieves the lowest path cost (98.27) among all algorithms while maintaining relatively fast discovery of the initial path (462 iterations). Although the computation time is longer than other algorithms due to additional optimization mechanisms,

the resulting trajectory is shorter and smoother. This result indicates that the hybrid sampling and adaptive rewiring mechanisms contribute to improved path refinement even in relatively simple environments.

Table 1. The performance of six different algorithms in environment 1

Algorithm	Path cost	Initial path found	Consumed time
RRT	101.2370	547	157.1268
RRT*	104.1222	921	149.222
RRT* Smart	99.5878	720	193.900
Fast RRT*	102.5661	428	132.3635
Informed-RRT*	98.7341	559	145.0513
Proposed method	98.2650	462	547.6206

3.3. Environment 2: medium-complexity map

Environment 2 introduces a higher level of difficulty with moderately distributed obstacles. This environment evaluates the ability of each algorithm to maintain exploration efficiency while avoiding obstacles during tree expansion. Figure 3 presents the visual comparison of path planning results in this environment. Figures 3(a)–(f) corresponds to the results of RRT, RRT*, Smart-RRT*, Fast-RRT*, Informed-RRT*, and ABH-RRT*, respectively.

In this scenario, several baseline algorithms fail to generate feasible paths within the allowed iterations due to inefficient exploration in constrained regions. Fast-RRT* successfully discovers a path with a cost of 223.64 after 2898 iterations. In contrast, ABH-RRT* finds a shorter path with a cost of 219.45 using significantly fewer iterations (1692), see Table 2.

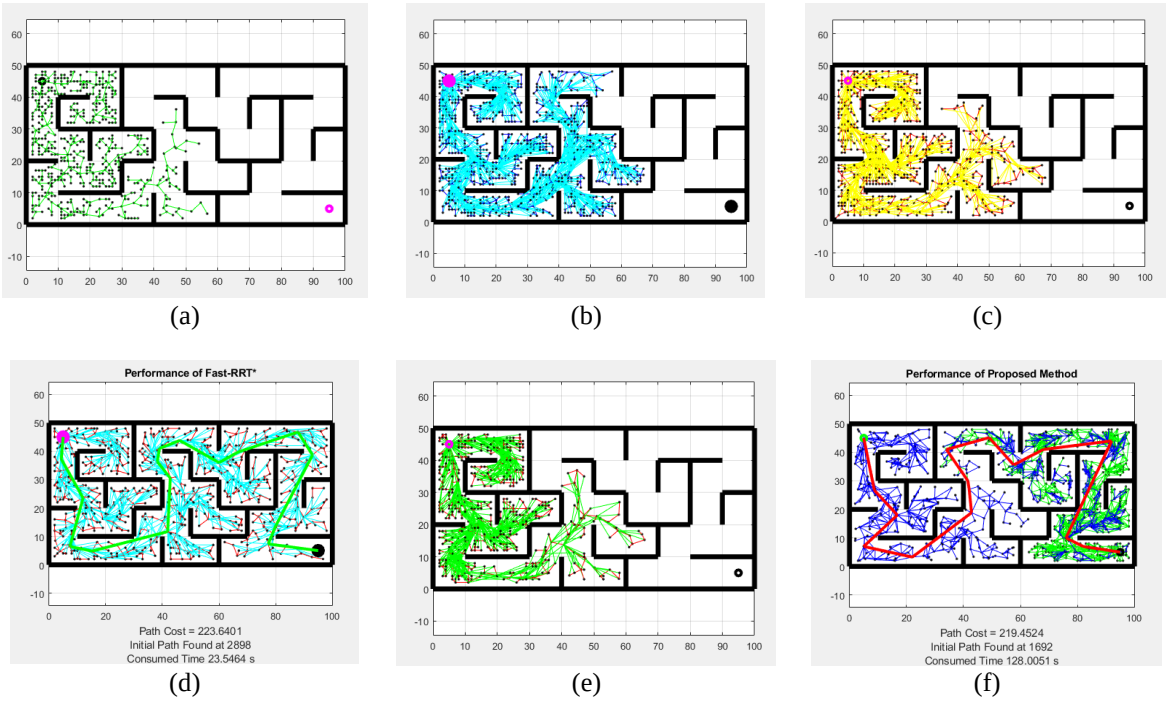

Figure 3. Path planning results of six algorithms in environment 2; (a) RRT, (b) RRT*, (c) Smart-RRT*, (d) Fast-RRT*, (e) Informed-RRT*, and (f) proposed method

Table 2. The performance of six different algorithms in environment 2

Algorithm	Path cost	Initial path found	Consumed time
RRT	Undefined	Unavailable	Undefined
RRT*	Undefined	Unavailable	Undefined
RRT* Smart	Undefined	Unavailable	Undefined
Fast RRT*	223.6401	2898	23.5464
Informed-RRT*	Undefined	Unavailable	Undefined
Proposed method	219.4524	1692	128.0051

This improvement can be attributed to the bidirectional exploration strategy, which increases the probability of connecting the start and goal regions earlier. Furthermore, adaptive parent selection improves local optimization of the trajectory after the initial path is discovered.

3.4. Environment 3: dense obstacle map

Environment 3 represents a challenging scenario with many obstacles, characterized by narrow paths and limited connections between the starting and ending points. These types of environments are known to be difficult for planners that use sampling methods, mainly because the space available for exploration is restricted.

Figures 4(a)–(f) illustrates representative paths produced by the evaluated algorithms. In this environment, traditional RRT produces the longest path because exploration is entirely random. RRT* improves path quality through rewiring, while Smart-RRT* further enhances convergence through guided sampling.

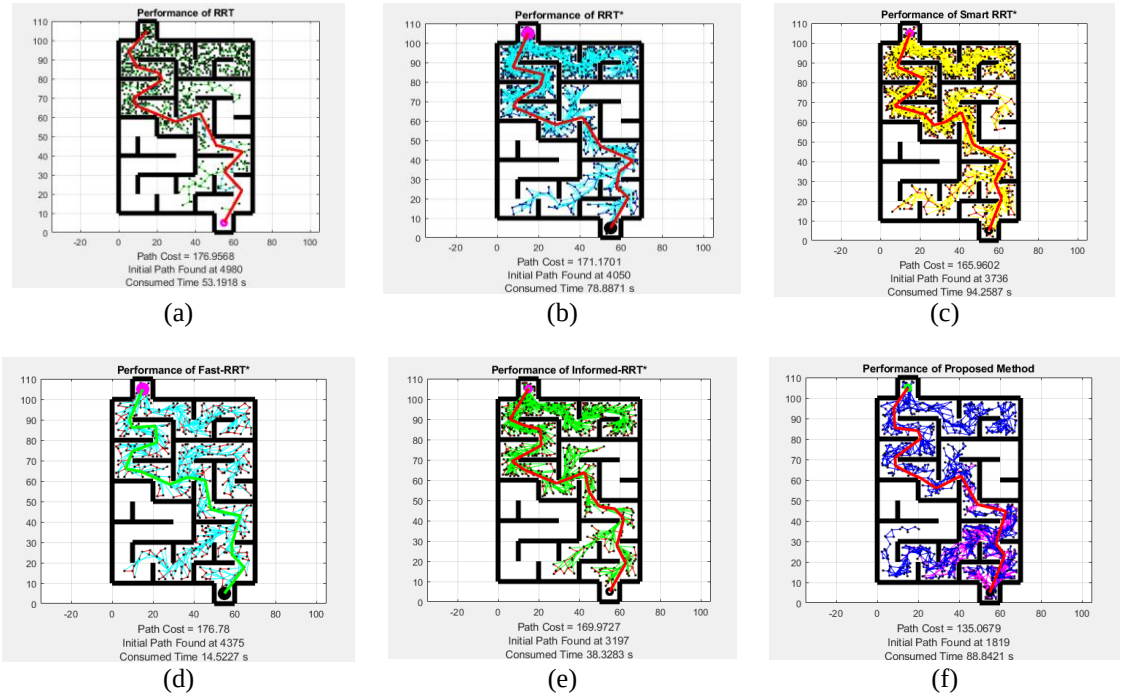

Figure 4. Path planning results of six algorithms in environment 3; (a) RRT, (b) RRT*, (c) Smart-RRT*, (d) Fast-RRT*, (e) Informed-RRT*, and (f) proposed method

However, as shown in Table 3, ABH-RRT* demonstrates the most significant improvement, producing the shortest trajectory with a path cost of 135.07 and requiring only 1819 iterations to reach the goal. Compared with Fast-RRT*, which requires 4375 iterations, the proposed method reduces the required iterations by more than 50%. This performance improvement is mainly due to the combination of bidirectional tree expansion and hybrid sampling, which effectively balances global exploration and local exploitation in complex obstacle distributions.

Table 3. The performance of six different algorithms in environment 3

Algorithm	Path cost	Initial path found	Consumed time
RRT	176.9568	4980	53.1918
RRT*	171.1701	4050	78.8871
RRT* Smart	165.9602	3736	94.2587
Fast RRT*	176.7800	4375	14.5227
Informed-RRT*	169.9727	3197	38.3283
Proposed method	135.0679	1819	88.8421

3.5. Environment 4: narrow corridor map

The final experiment evaluates algorithm performance in a highly constrained environment containing narrow corridors and limited connectivity. This scenario represents one of the most difficult cases for sampling-based path planning. Figures 5(a)–(f) show the path planning results of the six evaluated algorithms in Environment 4. The results indicate that RRT, RRT*, Smart-RRT*, and Informed-RRT* failed to generate feasible paths because they were unable to effectively explore the narrow passages in the environment.

Fast-RRT* generates a path with a cost of 334.10 after 9339 iterations. In contrast, the proposed ABH-RRT* algorithm finds a shorter path, with a cost of 317.31, and requires significantly fewer iterations (4949), as shown in Table 4.

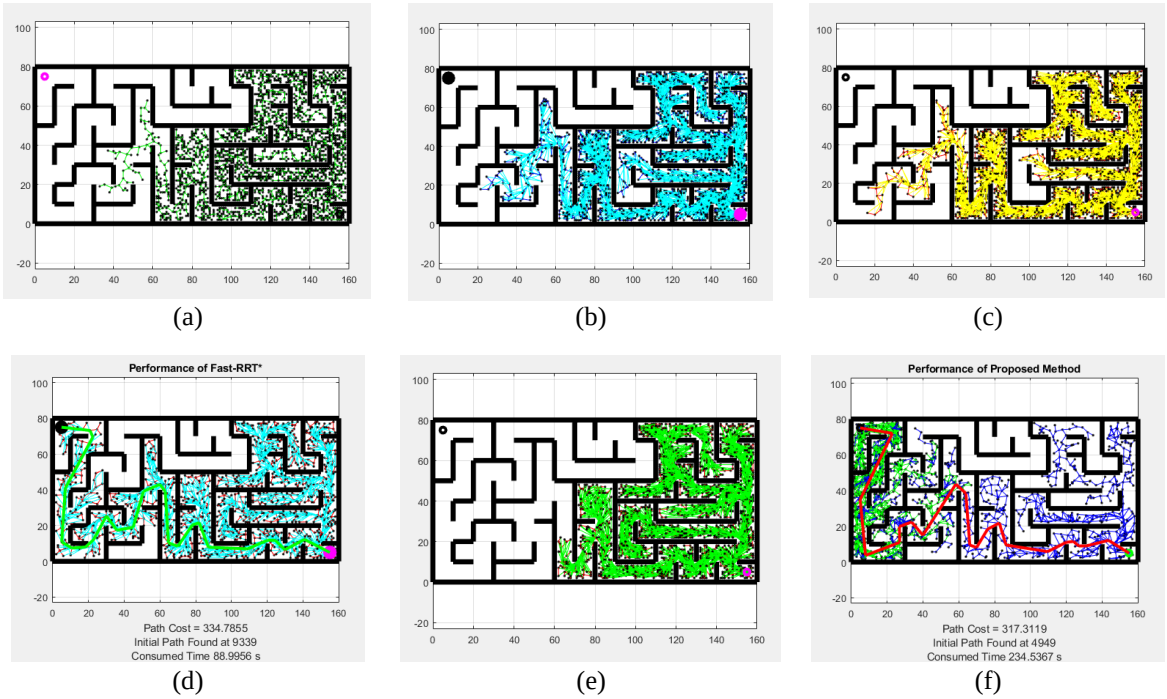

Figure 5. Path planning results of six algorithms in environment 4; (a) RRT, (b) RRT*, (c) Smart-RRT*, (d) Fast-RRT*, (e) Informed-RRT*, and (f) proposed method

Table 4. The performance of six different algorithms in environment 4

Algorithm	Path cost	Initial path found	Consumed time
RRT	Undefined	Unavailable	Undefined
RRT*	Undefined	Unavailable	Undefined
RRT* Smart	Undefined	Unavailable	Undefined
Fast RRT*	334.0992	9339	88.9956
Informed-RRT*	Undefined	Unavailable	Undefined
Proposed method	317.3119	4949	234.5367

In all four scenarios, the ABH-RRT* algorithm consistently improves both the quality of the paths and the speed of convergence. On average, the proposed method reduces path cost by 2–24% and increases convergence speed by 40–58% compared to the baseline algorithms. The performance improvements can be attributed to the integration of multiple complementary mechanisms. Bidirectional tree growth accelerates early exploration, hybrid sampling balances exploration and exploitation, and adaptive parent selection improves local optimization of the path structure. However, the additional optimization processes introduce higher computational overhead compared with simpler algorithms such as Fast-RRT*. This trade-off between computational cost and path quality should be considered when deploying the algorithm in real-time applications. In addition, the current evaluation focuses on two-dimensional static environments. Future work will extend the proposed approach to three-dimensional navigation scenarios and dynamic environments involving moving obstacles or multi-robot coordination.

4. CONCLUSION

This study proposed ABH-RRT*, a unified sampling-based path planning framework designed to improve exploration efficiency and path optimality in complex environments. The proposed method integrates four complementary mechanisms: bidirectional tree growth, adaptive parent selection, hybrid informed–fast sampling, and adaptive rewiring radius adjustment. These mechanisms work together to accelerate early exploration, improve local path optimization, and enhance convergence toward near-optimal trajectories. Experimental evaluations were conducted in four environments with different levels of complexity, including open spaces, moderately cluttered maps, dense obstacle environments, and narrow corridor scenarios. The results demonstrate that ABH-RRT* consistently outperforms conventional algorithms such as RRT, RRT*, Fast-RRT*, Smart-RRT*, and Informed-RRT*. In particular, the proposed method reduces path cost by 2–24% and improves convergence speed by 40–58% across the tested environments. The advantages of the algorithm become especially evident in dense obstacle maps and narrow corridor environments, where bidirectional exploration and hybrid sampling significantly improve the probability of finding feasible and optimized paths. Although the proposed approach improves path quality and convergence performance, the integration of multiple optimization mechanisms introduces additional computational overhead compared with simpler algorithms. Therefore, further optimization may be required for real-time deployment in embedded robotic systems.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the financial support provided by Universitas Mercu Buana (UMB).

FUNDING INFORMATION

This work was supported by Universitas Mercu Buana (UMB) under the Internal International Collaboration Scheme (2024–2025), Implementation Agreement No. 01-3-4/703/IA/KLN/IV/2025.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Heru Suwoyo	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			
Ahmad ‘Athif Mohd Faudzi		✓		✓		✓	✓	✓		✓		✓		
Andi Adriansyah	✓		✓		✓			✓		✓			✓	✓
Yudhi Gunardi		✓				✓				✓	✓		✓	
Julpri Andika			✓		✓					✓		✓	✓	✓
Yingzhong Tian		✓			✓		✓	✓		✓		✓		

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author, [initials: HS], upon reasonable request.

REFERENCES

[1] S. Ganesan and S. K. Natarajan, "A novel directional sampling-based path planning algorithm for ambient intelligence navigation scheme in autonomous mobile robots," *Journal of Ambient Intelligence and Smart Environments*, vol. 15, no. 3, pp. 269–284, Sep. 2023, doi: 10.3233/AIS-220292.

[2] M. Faroni, N. Pedrocchi, and M. Beschi, "Adaptive hybrid local–global sampling for fast informed sampling-based optimal path planning," *Autonomous Robots*, vol. 48, no. 2–3, p. 6, May 2024, doi: 10.1007/s10514-024-10157-5.

[3] X. Wang, Y. Feng, J. Tang, Z. Dai, and W. Zhao, "A UAV path planning method based on the framework of multi-objective jellyfish search algorithm," *Scientific Reports*, vol. 14, no. 1, p. 28058, Nov. 2024, doi: 10.1038/s41598-024-79323-0.

[4] H. Xu *et al.*, "ERRT-GA: Expert Genetic Algorithm with Rapidly Exploring Random Tree Initialization for Multi-UAV Path Planning," *Drones*, vol. 8, no. 8, p. 367, Aug. 2024, doi: 10.3390/drones8080367.

[5] X. Zhou, G. Shi, and J. Zhang, "Improved Grey Wolf Algorithm: A Method for UAV Path Planning," *Drones*, vol. 8, no. 11, p. 675, Nov. 2024, doi: 10.3390/drones8110675.

[6] Q. Zhou and G. Liu, "UAV Path Planning Based on the Combination of A-star Algorithm and RRT-star Algorithm," in *2022 IEEE International Conference on Unmanned Systems (ICUS)*, Guangzhou, China: IEEE, Oct. 2022, pp. 146–151. doi: 10.1109/ICUS55513.2022.9986703.

[7] 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, p. 7898, Nov. 2021, doi: 10.3390/s21237898.

[8] J. Qi, Q. Yuan, C. Wang, X. Du, F. Du, and A. Ren, "Path planning and collision avoidance based on the RRT*FN framework for a robotic manipulator in various scenarios," *Complex & Intelligent Systems*, vol. 9, no. 6, pp. 7475–7494, Dec. 2023, doi: 10.1007/s40747-023-01131-2.

[9] S. Fu, D. Yang, Z. Mei, and W. Zheng, "Progress in Construction Robot Path-Planning Algorithms: Review," *Applied Sciences*, vol. 15, no. 3, p. 1165, Jan. 2025, doi: 10.3390/app15031165.

[10] X. Jiang, Z. Wang, and C. Dong, "A Path Planning Algorithm Based on Improved RRT Sampling Region," *Computers, Materials & Continua*, vol. 80, no. 3, pp. 4303–4323, 2024, doi: 10.32604/cmc.2024.054640.

[11] F. S. Elkazzaz, M. A. H. Abozied, and C. Hu, "Hybrid RRT/DE algorithm for high performance UCAV path planning," in *ACM International Conference Proceeding Series*, pp. 235–242, 2017, doi: 10.1145/3171592.3171618.

[12] S. Klemm *et al.*, "RRT*-Connect: Faster, asymptotically optimal motion planning," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Zhuhai, China: IEEE, Dec. 2015, pp. 1670–1677. doi: 10.1109/ROBIO.2015.7419012.

[13] X. Wang, J. Wei, X. Zhou, Z. Xia, and X. Gu, "AEB-RRT*: an adaptive extension bidirectional RRT* algorithm," *Autonomous Robots*, vol. 46, no. 6, pp. 685–704, Aug. 2022, doi: 10.1007/s10514-022-10044-x.

[14] J. Chen, Y. Zhao, and X. Xu, "Improved RRT-Connect Based Path Planning Algorithm for Mobile Robots," *IEEE Access*, vol. 9, pp. 145988–145999, 2021, doi: 10.1109/ACCESS.2021.3123622.

[15] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan, "RRT*-Smart: Rapid convergence implementation of RRT* towards optimal solution," in *2012 IEEE International Conference on Mechatronics and Automation, ICMA 2012*, pp. 1651–1656, 2012, doi: 10.1109/ICMA.2012.6284384.

[16] J. Nasir *et al.*, "RRT*-SMART: A rapid convergence implementation of RRT*," *International Journal of Advanced Robotic Systems*, vol. 10, 2013, doi: 10.5772/56718.

[17] H.-T. Tak, C.-G. Park, and S.-C. Lee, "Improvement of RRT*-Smart Algorithm for Optimal Path Planning and Application of the Algorithm in 2 & 3-Dimension Environment," *Journal of the Korean Society for Aviation and Aeronautics*, vol. 27, no. 2, pp. 1–8, Jun. 2019, doi: 10.12985/ksaa.2019.27.2.001.

[18] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT∗: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *IEEE International Conference on Intelligent Robots and Systems*, pp. 2997–3004, 2014, doi: 10.1109/IROS.2014.6942976.

[19] L. Yuan, J. Zhao, W. Li, and J. Hou, "Improved Informed-RRT* Based Path Planning and Trajectory Optimization for Mobile Robots," *International Journal of Precision Engineering and Manufacturing*, vol. 24, no. 3, pp. 435–446, 2023, doi: 10.1007/s12541-022-00756-6.

[20] D. Wu, L. Wei, G. Wang, L. Tian, and G. Dai, "APF-IRRT*: An Improved Informed Rapidly-Exploring Random Trees-Star Algorithm by Introducing Artificial Potential Field Method for Mobile Robot Path Planning," *Applied Sciences (Switzerland)*, vol. 12, no. 21, 2022, doi: 10.3390/app122110905.

[21] Q. Li, J. Wang, H. Li, B. Wang, and C. Feng, "Fast-RRT*: An Improved Motion Planner for Mobile Robot in Two-Dimensional Space," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 17, no. 2, pp. 200–208, 2022, doi: 10.1002/tee.23502.

[22] Z. Wu, Z. Meng, W. Zhao, and Z. Wu, "Fast-RRT: A RRT-based optimal path finding method," *Applied Sciences (Switzerland)*, vol. 11, no. 24, 2021, doi: 10.3390/app112411777.

[23] A. Adriansyah, N. Ferdana, S. Budiyanto, and J. Andika, "Design of Telemedicine Robot using Behavior-based Control Architecture with Two-Step Fuzzy Logic Optimization," *Journal of Computer Science*, vol. 15, no. 11, pp. 1617–1626, Nov. 2019, doi: 10.3844/jcssp.2019.1617.1626.

[24] Z. Zhang, X. Bai, H. Yang, and S. Zhang, "Dynamic Path Planning via Enhanced ACO and DWA Algorithms," *Applied Sciences*, vol. 16, no. 2, p. 583, Jan. 2026, doi: 10.3390/app16020583.

[25] Z. Wang, G. Sun, K. Zhou, and L. Zhu, "A parallel particle swarm optimization and enhanced sparrow search algorithm for unmanned aerial vehicle path planning," *Heliyon*, vol. 9, no. 4, p. e14784, Apr. 2023, doi: 10.1016/j.heliyon.2023.e14784.

[26] D. K. Muhsen, F. A. Raheem, and A. T. Sadiq, "Improved rapidly exploring random tree using salp swarm algorithm," *Journal of Intelligent Systems*, vol. 33, no. 1, p. 20230219, Mar. 2024, doi: 10.1515/jisys-2023-0219.

[27] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime Motion Planning using the RRT*," in *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China: IEEE, May 2011, pp. 1478–1483. doi: 10.1109/ICRA.2011.5980479.

[28] A. Perez, R. Platt, G. Konidaris, L. Kaelbling, and T. Lozano-Perez, "LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics," in *2012 IEEE International Conference on Robotics and Automation*, St Paul, MN, USA: IEEE, May 2012, pp. 2537–2542. doi: 10.1109/ICRA.2012.6225177.

[29] X. Jin, Z. Yan, H. Yang, Q. Wang, and G. Yin, "A Goal-Biased RRT Path Planning Approach for Autonomous Ground Vehicle," in *2020 4th CAA International Conference on Vehicular Control and Intelligence (CVCI)*, Hangzhou, China: IEEE, Dec. 2020, pp. 743–746. doi: 10.1109/CVCI51460.2020.9338597.

[30] C. Urmson and R. Simmons, "Approaches for heuristically biasing RRT growth," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, Las Vegas, NV, USA: IEEE, 2003, pp. 1178–1183. doi: 10.1109/IROS.2003.1248805.

[31] X. Wang, "Genetic RRT: Asymptotically Optimal Sampling-based Path Planning via Optimization of Genetic Algorithm," *Highlights in Science, Engineering and Technology*, vol. 43, pp. 215–222, Apr. 2023, doi: 10.54097/hset.v43i.7423.
[32] C.-H. Wei and J.-S. Liu, "Hybridizing RRT and variable-length genetic algorithm for smooth path generation," in *2011 IEEE International Conference on Robotics and Biomimetics*, Karon Beach, Thailand: IEEE, Dec. 2011, pp. 626–632. doi: 10.1109/ROBIO.2011.6181356.
[33] J. Ma, Y. Liu, S. Zang, and L. Wang, "Robot Path Planning Based on Genetic Algorithm Fused with Continuous Bezier Optimization," *Computational Intelligence and Neuroscience*, vol. 2020, pp. 1–10, Feb. 2020, doi: 10.1155/2020/9813040.
[34] H. T. Najm, N. S. Ahmad, and A. S. Al-Araji, "Enhanced path planning algorithm via hybrid WOA-PSO for differential wheeled mobile robots," *Systems Science & Control Engineering*, vol. 12, no. 1, p. 2334301, Dec. 2024, doi: 10.1080/21642583.2024.2334301.
[35] B. Song, Z. Wang, and L. Zou, "An improved PSO algorithm for smooth path planning of mobile robots using continuous high-degree Bezier curve," *Applied Soft Computing*, vol. 100, p. 106960, Mar. 2021, doi: 10.1016/j.asoc.2020.106960.
[36] Y. Wang, J. Cao, X. Zheng, Y. Zhang, Y. Zhang, and W. Chen, "Path planning of RRT* algorithm with subregional dynamic probabilistic sampling based on artificial potential field in radiation environments," *Nuclear Engineering and Technology*, vol. 57, no. 10, p. 103706, Oct. 2025, doi: 10.1016/j.net.2025.103706.
[37] A. Viseras, R. O. Losada, and L. Merino, "Planning with ants: Efficient path planning with rapidly exploring random trees and ant colony optimization," *International Journal of Advanced Robotic Systems*, vol. 13, no. 5, p. 1729881416664078, Sep. 2016, doi: 10.1177/1729881416664078.
[38] D. K. Muhsen, F. A. Raheem, Y. Yusof, A. T. Sadiq, and F. Al Alawy, "Improved Rapidly-Exploring Random Tree using Firefly Algorithm for Robot Path Planning," *Journal of Soft Computing and Computer Applications*, vol. 1, no. 2, Dec. 2024, doi: 10.70403/3008-1084.1009.
[39] S. Hosseininejad and C. Dadkhah, "Mobile robot path planning in dynamic environment based on cuckoo optimization algorithm," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, p. 1729881419839575, Mar. 2019, doi: 10.1177/1729881419839575.
[40] A. Husaković, L. Banjanović-Mehmedović, and T. Konjić, "Efficiency Boost: Service Robot Path Planning with Grey Wolf Optimization," in *2024 23rd International Symposium INFOTEH-JAHORINA (INFOTEH)*, East Sarajevo, Bosnia and Herzegovina: IEEE, Mar. 2024, pp. 1–6. doi: 10.1109/INFOTEH60418.2024.10495949.
[41] R. Kumar, L. Singh, and R. Tiwari, "Path planning for the autonomous robots using modified grey wolf optimization approach," *Journal of Intelligent & Fuzzy Systems*, vol. 40, no. 5, pp. 9453–9470, Apr. 2021, doi: 10.3233/JIFS-201926.
[42] B. Tu, F. Wang, X. Han, and X. Fu, "Q-learning Guided Grey Wolf Optimizer for UAV 3D Path Planning," *International Journal of Advanced Computer Science and Applications*, vol. 15, no. 7, 2024, doi: 10.14569/IJACSA.2024.0150747.
[43] A. A. Alabdalbari and I. A. Abed, "New robot path planning optimization using hybrid GWO-PSO algorithm," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 3, pp. 1289–1296, Jun. 2022, doi: 10.11591/eei.v11i3.3677.
[44] I. B. Jeong, S. J. Lee, and J. H. Kim, "Quick-RRT*: Triangular inequality-based implementation of RRT* with improved initial solution and convergence rate," *Expert Systems with Applications*, vol. 123, pp. 82–90, 2019, doi: 10.1016/j.eswa.2019.01.032.
[45] X. Cui, C. Wang, Y. Xiong, L. Mei, and S. Wu, "More Quickly-RRT*: Improved Quick Rapidly-exploring Random Tree Star algorithm based on optimized sampling point with better initial solution and convergence rate," *Engineering Applications of Artificial Intelligence*, vol. 133, p. 108246, Jul. 2024, doi: 10.1016/j.engappai.2024.108246.
[46] J. Cong, J. Hu, Y. Wang, Z. He, L. Han, and M. Su, "FF-RRT*: a sampling-improved path planning algorithm for mobile robots against concave cavity obstacle," *Complex & Intelligent Systems*, vol. 9, no. 6, pp. 7249–7267, Dec. 2023, doi: 10.1007/s40747-023-01111-6.
[47] L. Zhu, P. Duan, L. Meng, and X. Yang, "GAO-RRT*: A path planning algorithm for mobile robot with low path cost and fast convergence," *AIMS Mathematics*, vol. 9, no. 5, pp. 12011–12042, 2024, doi: 10.3934/math.2024587.

BIOGRAPHIES OF AUTHORS

Heru Suwoyo is a lecturer in the Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana. He obtained his Dr.Eng. degree in Mechatronic Engineering from Shanghai University. His research interests focus on perception and navigation of robotic systems, including path planning, path tracking, behavior-based movement, and visual odometry, simultaneous localization, and mapping. He can be contacted at email: heru.suwoyo@mercubuana.ac.id.

Ahmad 'Athif Mohd Faudzi (Senior Member, IEEE) is a Professor in the Faculty of Electrical Engineering, Universiti Teknologi Malaysia, and a Research Fellow at the Centre for Artificial Intelligence and Robotics (CAIRO). He received his B.Eng. degree in Computer Engineering and M.Eng. degree in Mechatronics and Automatic Control from Universiti Teknologi Malaysia in 2004 and 2006, respectively, and his Dr.Eng. degree in System Integration from Okayama University, Japan, in 2010. His research interests focus on soft actuators, inspection robotics, and bio-inspired robotics. He can be contacted at email: athif@utm.my.

Andi Adriansyah is lecturer at Department of Electrical Engineering, Universitas Mercu Buana, Indonesia. He obtained Ph.D. degree in Electrical Engineering from Universitas Teknologi Malaysia, Malaysia in 2007. His research interest includes mobile robot navigation, soft robotics, and closedloop controllers such as PID and FL controllers, heuristic algorithm-based tuning method, and artificial intelligence. He can be contacted at email: andi@mercubuana.ac.id.

Yudhi Gunardi is an Associate Professor in the Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana. He obtained his Ph.D. degree from the Department of Mechatronic and Robotic Engineering, Universiti Tun Hussein Onn Malaysia. His research interests include information systems, automation using Arduino, electronics with Arduino, Arduino projects, and mobile robotics. He can be contacted at email: yudhi.gunardi@mercubuana.ac.id.

Julpri Andika is a lecturer in the Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana (UMB). He received his S.T. degree from Universitas Mercu Buana and his M.Sc. degree from Beijing Institute of Technology (BIT), specializing in control and robotics. His research and teaching interests include computer programming, digital systems, basic control systems, and robotics. He can be contacted at email: julpri.andika@mercubuana.ac.id.

Yingzhong Tian is a Professor in School of Mechatronic Engineering and Automation, and he is also the Director of Research in Lab of Intelligent Mechanism and Advanced Robot (LIMAR) research team at Shanghai University. He obtained a Ph.D. degree in Mechanical manufacture and automation in 2007 from Shanghai University. His research interests include mobile robots, bionic robots, multi-robot systems, and image processing. He can be contacted at email: troytian@shu.edu.cn.