

Research Article

Open Access



A node selection algorithm to graph-based multi-waypoint optimization navigation and mapping

Timothy Sellers¹, Tingjun Lei¹, Chaomin Luo¹, Gene Eu Jan², Junfeng Ma³

¹Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS 39759, USA.

²Department of Electrical Engineering, National Taipei University and Tainan National University of the Arts, Taipei 72045, Taiwan.

³Department of Industrial and Systems Engineering, Mississippi State University, Mississippi State, MS 39759, USA.

Correspondence to: Prof. Chaomin Luo, Department of Electrical and Computer Engineering, Mississippi State University, 406 Hardy Road, Mississippi State, MS 39762, USA. E-mail: Chaomin.Luo@ece.msstate.edu; ORCID: 0000-0002-7578-3631

How to cite this article: Sellers T, Lei T, Luo C, Jan GE, Ma J. A node selection algorithm to graph-based multi-waypoint optimization navigation and mapping. *Intell Robot* 2022;2(4):333-54. <http://dx.doi.org/10.20517/ir.2022.21>

Received: 20 Jul 2022 **First Decision:** 12 Aug 2022 **Revised:** 18 Aug 2022 **Accepted:** 25 Aug 2022 **Published:** 12 Oct 2022

Academic Editor: Simon X. Yang **Copy Editor:** Jia-Xin Zhang **Production Editor:** Jia-Xin Zhang

Abstract

Autonomous robot multi-waypoint navigation and mapping have been demanded in many real-world applications found in search and rescue (SAR), environmental exploration, and disaster response. Many solutions to this issue have been discovered via graph-based methods in need of switching the robot's trajectory between the nodes and edges within the graph to create a trajectory for waypoint-to-waypoint navigation. However, studies of how waypoints are locally bridged to nodes or edges on the graphs have not been adequately undertaken. In this paper, an adjacent node selection (ANS) algorithm is developed to implement such a protocol to build up regional path from waypoints to nearest nodes or edges on the graph. We propose this node selection algorithm along with the generalized Voronoi diagram (GVD) and Improved Particle Swarm Optimization (IPSO) algorithm as well as a local navigator to solve the safety-aware concurrent graph-based multi-waypoint navigation and mapping problem. Firstly, GVD is used to form a Voronoi diagram in an obstacle populated environment to construct safety-aware routes. Secondly, the sequence of multiple waypoints is created by the IPSO algorithm to minimize the total travelling cost. Thirdly, while the robot attempts to visit multiple waypoints, it traverses along the edges of the GVD to plan a collision-free trajectory. The regional path from waypoints to the nearest nodes or edges needs to be created to join the trajectory by the proposed ANS algorithm. Finally, a sensor-based histogram local reactive navigator is adopted for moving obstacle avoidance while local maps are constructed as the robot moves. An improved \mathcal{B} -spline curve-based smooth scheme is adopted that further refines the trajectory and enables the robot to be navigated smoothly. Simulation and comparison studies validate the effectiveness and robustness of the proposed model.



© The Author(s) 2022. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License (<https://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, sharing, adaptation, distribution and reproduction in any medium or format, for any purpose, even commercially, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.



Keywords: Adjacent node selection (ANS) algorithm, safety-aware roads, path planning, multiple-waypoint optimization, navigation and mapping

1. INTRODUCTION

Robotics system has been applied to numerous fields, such as transportation^[1], healthcare service^[2,3], agriculture^[4], manufacturing^[5], *etc.*, in recent years. Robot navigation is one of the fundamental components in robotic systems, which includes multi-waypoint navigation system^[6–8]. As an increasing demand and limited onboard resources for autonomous robots, it requires the ability to visit several targets in one mission to optimize multiple objectives, including time, robot travel distance minimization, and spatial optimization^[9–15]. For example, due to a global pandemic, the world struggled to sanitize heavily populated areas, such as airports, hospitals, and educational buildings. Autonomous robots with multi-waypoint navigation systems can effectively sanitize all targeted areas without endangering the workers^[14,16]. As well as in agriculture management, multi-waypoint strategies allow the robotic system to navigate and survey multiple areas to assist in production and collection.

In order to employ robotic systems in real-world scenarios, one critical factor is to develop autonomous robot multi-waypoint navigation and mapping system^[17]. In order to solve the autonomous robot navigation problem, countless algorithms have been developed, such as graph-based^[18,19], ant colony optimization (ACO)^[20–22], bat-pigeon algorithm (BPA)^[23], neural networks^[24–26], fuzzy logic^[27], artificial potential field (APF)^[28], sampling-based strategy^[14,29], hybrid algorithms^[30], task planning algorithm^[31], *etc.* Chen *et al.* produced a hybrid graph-based reinforcement learning architecture to develop a method for robot navigation in crowds^[18]. Luo *et al.* proposed an improved vehicle navigation method, which utilizes a heading-enabled ACO algorithm to improve trajectory towards the target^[20]. Lei *et al.* developed a Bat-Pigeon algorithm with the ability to adjust the speed navigation of autonomous vehicles^[23]. Luo *et al.* developed the model for multiple robots complete coverage navigation while using a bio-inspired neural network to dynamically avoid obstacles^[32]. Na and Oh established a hybrid control system for autonomous mobile robot navigation that utilizes a neural network for environment classification and behavior-based control method to mimic the human steering commands^[25]. Lazreg and Benamrane^[27] developed a neuro-fuzzy inference system associated with a Particle Swarm Optimization (PSO) method for robot path planning using a variety of sensors to control the speed and position of a robot. Jensen-Nau *et al.*^[28] integrated a Voronoi-based path generation algorithm and an artificial potential field path planning method, in which the latter is capable of establishing a path in an unknown environment in real-time for robot path planning and obstacle avoidance. Penicka and Scaramuzza^[14] developed a sampling-based multi-waypoint minimum-time path planning model that allows obstacle avoidance in cluttered environments. Ortiz and Yu^[30] proposed a sliding control method in combination with simultaneous localization and mapping (SLAM) method to overcome the bounded uncertainties problem, which utilizes a genetic algorithm to improve path planning capabilities. Bernardo *et al.* proposed a task planning method for home environment ontology to translate tasks given by other robots or humans into feasible tasks for another robot agent^[31].

In robotic path planning, one of the special topics is autonomous robot multi-waypoint navigation which has been studied for many years. For instance, Shair *et al.* proposed a model for real-world waypoint navigation using a variety of sensors for accurate environmental analysis^[33]. The system is designed utilizing the wide area augmentation system (WAAS) and the European geostationary navigation overlay service (EGNOS) for GPS in combination with aerial images to provide valuable positioning data to the system. Yang^[34] brought about a multi-waypoint navigation system based on terrestrial signals of opportunity (SOPs) transmitters, which has the ability to operate in environments that are not available to global navigation satellite systems (GNSS) for unmanned aerial vehicles (UAV). Janoš *et al.* proposed a sampling-based multi-waypoint path planner, space-

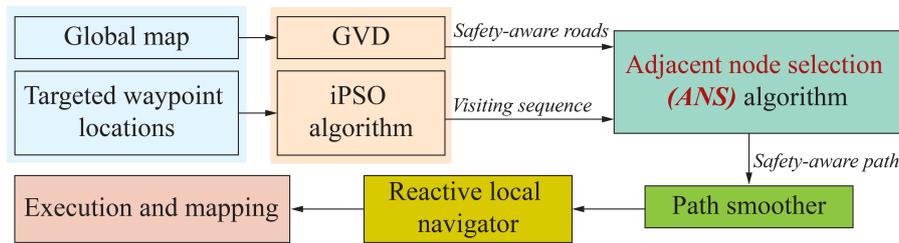


Figure 1. Illustration of the proposed model framework.

filling forest method to solve the problem of finding collision-free trajectories while the sequence of waypoints is formed by multiple trees^[29]. However, the aforementioned approaches have not taken into account the safety of the robot during navigation. In real-world applications, an autonomous vehicle has odometry errors during operation. The safety-aware road model is developed utilizing the Generalized Voronoi diagram (GVD) approach. Once the safety-aware roads are defined, a particle swarm optimization (PSO) algorithm-based multi-waypoint path planning algorithm is proposed to visit each waypoint in an explicated sequence while simultaneously avoiding obstacles. In this paper, the Adjacent Node Selection (ANS) algorithm is developed to select the closest nodes on the safety-aware roads to generate the final collision-free trajectories with minimal distance. Furthermore, in our hybrid algorithm, we utilize a histogram-based reactive local navigator to avoid dynamic and unknown obstacles within the workspace. Through all of these methods, an effective and efficient safety-aware multiple waypoint navigation model was established, which has been validated by both simulations and comparison studies.

This paper proposes an Adjacent Node Selection (ANS) algorithm for obtaining an optimal access node into graph-based maps. To the best of our knowledge, there are no known similar algorithms that improve the paths created from the waypoint to the graph. The ANS algorithm can be applied to any graph-based mapping environment, which improves the various graph-based models used for autonomous robotic path planning systems. In finding an access point into the graph utilizing one of the nodes in the system, the ANS algorithm conducts point-to-point selection in dense obstacle field of environments to obtain a node to gain access to the graph that forms a resumable path from a waypoint to the graph. The algorithm's overall goal is to find a node in a graph-based map and shorten the overall path length from each waypoint to waypoint, and the waypoint to the graph.

One can see in [Figure 1](#) the overall framework of our proposed model. Initially, the model is provided with a global map of its environment and the location of each target waypoint. The GVD utilizes the global map to construct the safety-aware roads, which are used to guide the robot throughout the environment. The targeted waypoint locations are used as input to the improved PSO (IPSO) algorithm that act as our waypoint sequencing module, which is utilized to find a near-optimal sequence to visit each waypoint. The ANS algorithm utilizes the output from the previous stages. The ANS algorithm finds the best node for each waypoint to use as an access point to the graph. If there is no direct path from one waypoint to a node in the graph, the algorithm conduces point-to-point navigation with nodes within a specified range, which will be explained in later sections of the paper. Once each waypoint has found its access point, the safety-aware path is constructed, which is then applied to our path smoothing algorithm. Finally, we utilize a reactive local navigation system to detect obstacles autonomously and simultaneously build a map of the environment along the generated path.

The main contributions of this paper with this framework of concurrent multi-waypoint navigation and mapping with collision avoidance as are summarized as follows: (1) An adjacent node selection (ANS) algorithm is proposed to build up regional bridges from waypoints to nodes or edges on the graph in multi-waypoint navigation and mapping; (2) a concurrent multi-waypoint navigation and mapping framework of an autonomous robot is developed by the generalized Voronoi diagram (GVD) and IPSO algorithm as well as a local navigator;

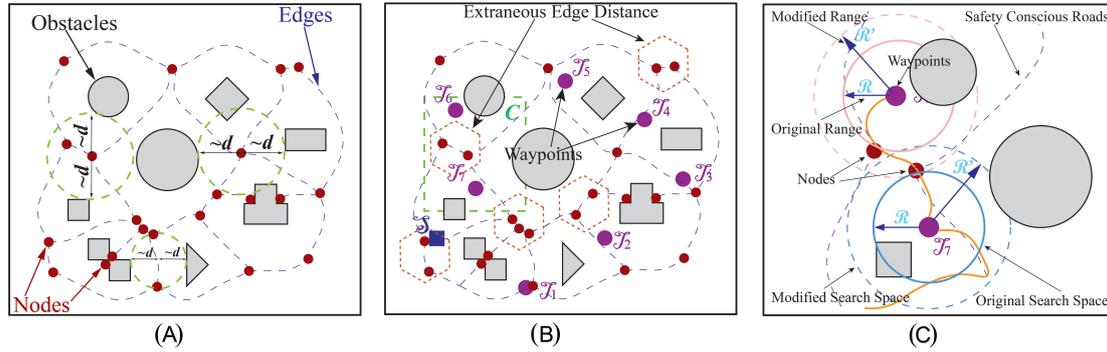


Figure 2. Illustration of the ANS algorithm and the safety-aware model by creation of the GVD. (A) Depicts the equidistance property of the nodes and edges in the GVD model. (B) illustrates how we remove extraneous edge distances, which are small edges within the graph that reduce the search space or range in the ANS algorithm. (C) illustrates how the removal of those extraneous edges improves the range and how an access node to the graph can be found.

(3) a GVD method is employed to plan safety-aware trajectories in an obstacle populated environment, while the sequence of multiple waypoints is created by the IPSO algorithm to minimize the total distance cost; (4) a sensor-based histogram robot reactive navigator coupled with map building is adopted for moving obstacle avoidance and local mapping. An improved \mathcal{B} -spline curve-based speed modulation module is adopted for smoother navigation.

The structure of this paper is as follows. Section II presents the safety-aware model that constructs the safety-conscious roads for robot traversal. Section III describes the proposed adjacent node selection algorithm to find an access node into the graph-based map. Section IV, the improved PSO-based (IPSO) multi-waypoint navigation model for waypoint sequencing is explained. Section V illustrates the reactive local navigator for real-time workspace building and obstacle avoidance. Section VI depicts simulation results and comparative analyses. Several important properties of the proposed framework are summarized in Section VII.

2. SAFETY-AWARE MODEL

Computational geometry has exceedingly studied the Voronoi diagrams (VD) model, which is an elemental data structure used as a minimizing diagram of a finite set of continuous functions^[28,35]. The minimized function defines the distance to an object within the workspace. The VD model decomposes the workspace into several regions, each consisting of the points near a given object than the others. Let $\mathcal{G} = \{g_1 \cdots g_n\}$ be a set of points of \mathbb{R}^d to each g_i associated with a specific Voronoi region $V(g_i)$. This can be expressed by the following equation^[36,37]:

$$V(g_i) = \{x \in \mathbb{R}^d : \|x - g_i\| \leq \|x - g_j\|, \forall j \leq n\} \quad (1)$$

The intersection of $n - 1$ half spaces can be denoted by the region $V(g_i)$. Each half space holds a point g_i along with another point of \mathcal{G} . The regions $V(g_i)$ are convex polyhedrons due to the bisectors acting as hyperplanes between each region. The generalized Voronoi diagram (GVD) is a modified version of the VD model defined as the set of points Euclidean distance from two obstacles^[37]. The workspace is represented as a graph by the GVD model consisting of nodes, edges, and vertices^[28].

The proposed ANS algorithm utilizes nodes and edges obtained from the GVD illustrated in [Figure 2A](#). The nodes in the GVD act as junctions to connect waypoints, whereas the edges are used to determine the search range in view of waypoints. Solely the nodes in the search range are calculated, other than the entire workspace, thus reducing the computational expense. [Figure 2B](#) reveals that the range of some edges fails to evaluate how spare of the workspace is configured. These edges with short lengths are distractors for evaluating and computing the search range, thus need to be eliminated shown in [Figure 2B](#). In our ANS algorithm, those

edges with the shortest 10% are eliminated, while the rest of the edges are averaged to compute the radius of the search range. One circumstance is exhibited in Figure 2C, in which some nodes fail to fall into the search range once the excessively short edges are excluded. The original search space in solid circles and modified search space in dashed circles based on the ANS algorithm are shown in Figure 2C. As may be seen, after excessively short edges are effectively eliminated, an appropriate search range is achieved.

GVD nodes form the Euclidean distance between two or more obstacles, while the edges are the junction of two nodes that depict the distance between each neighboring node to another. Vertices are the connection points between three or more nodes. Using these features from the GVD model, an obstacle-free path with our safety-aware model is effectively created. The safety-aware model is constructed by inputting an image and extracting all significant features from the image, thus allowing the model to construct a map from the input image. The safety-aware roads are the clearest path between obstacles that occupy the available space in the map, which can be seen in Figure 2.

In this paper, in order to explain our proposed ANS algorithm, how the nodes are determined and constructed with the GVD will be presented in some detail. We will introduce some definitions and notations. Lee and Drysdale^[36] derive four basic definitions for determining the optimal placement of edges and nodes within the GVD graph.

DEFINITION 1. A closed line segment M consists of two endpoints α and γ . A straight line is denoted by (α, γ) , which is also known as an open segment. Elements within the derivation are referred to as points or segments. The straight line containing M is denoted by \overleftrightarrow{M} . The same line directed from α to γ is denoted by \vec{M} .

DEFINITION 2. The projection $\sigma(\epsilon, M)$ of a point q onto a closed segment M , is the intersection of \overleftrightarrow{M} and is perpendicular to M and passing through ϵ .

DEFINITION 3. The distance between $\omega(\epsilon, M)$ is a point ϵ and a closed segment M in the Euclidean metric is defined as the distance $\omega(\epsilon, \sigma(\epsilon, M))$ between the point ϵ and its projection onto M if $\sigma(\epsilon, M)$ belongs to M and is $\min(\omega(\epsilon, \alpha), \omega(\epsilon, \gamma))$ otherwise. In other words, $\omega(\epsilon, M) = \min_{u \in M}(\epsilon, u)$. The point of M , which is closest to ϵ , is called the image $I(\epsilon, M)$ of ϵ on M .

DEFINITION 4. The bisector $\beta(c_i, c_j)$ of two elements c_i and c_j is the locus of points equidistant from c_i and c_j . The bisector $\beta(Z, Q)$ of two sets of elements Z and Q is defined to be the locus of points equidistant from Z and Q , where the distance $\omega(\epsilon, Z)$ between a point ϵ and a set of elements Z is defined to be $\min_{c \in Z} \omega(\epsilon, c)$. The bisector $\beta(c_i, c_j)$ is said to be oriented if a direction is imposed upon it so that elements c_i and c_j lie to the left and the right of it, respectively. An oriented bisector $\beta(Z, Q)$ is defined similarly.

Utilizing these four definitions, we can easily establish edges among spaces or obstacles, which uses the equidistant to create an optimal edge that lies directly between the two spaces. As seen in the above definitions, we can also create edges between irregular-shaped spaces and obstacles.

To characterize the GVD, we expect the robot to be operated at a point within the workspace, a W , which is populated by convex obstacles C_1, \dots, C_n . Non-convex obstacles are displayed as the union of convex shapes. The distance between a point and an obstacle is the minimal distance between the point and all points of the obstacle. The distance function, and its “gradient” are represented as:

$$d_i(x) = \min_{c_0 \in C_i} \|x - c_0\| \quad (2)$$

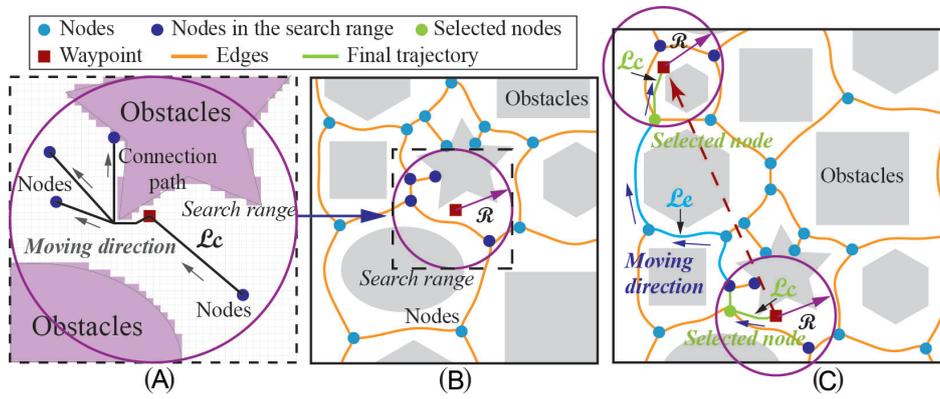


Figure 3. Details of the Adjacent Node Selection (ANS) algorithm. (A) The IPSO-based connection path planning in the search range. (B) The search range determination and node selection inside. (C) The final generated path with the minimum path length.

$$\nabla d_i(x) = \frac{x - c_0}{\|x - c_0\|} \tag{3}$$

where in Equation (2) d_i is the distance to obstacle C_i from a point x , and in Equation (3) $\nabla d_i(x)$ is the unit vector in the direction from x to c_0 , where c_0 is the closest point to x in C_i . The essential structure block of the GVD is the arrangement of points equidistant to two sets C_i and C_j , with the end goal that each set in this set is the minimal distance to the obstacles C_i and C_j than other obstacles. This type of structure is known as the two-equidistant face,

$$f(i, j) = \begin{cases} x \in \mathbb{R}^m : 0 \leq d_i(x) = d_j(x) \\ \forall d_i \neq i, j \\ \nabla d_i(x) \neq \nabla d_j(x) \end{cases} \tag{4}$$

Each face has a co-dimension in the ambient space, which causes the two-equidistant faces to be seen as one-dimensional. The intersection of both faces forms the GVD and is denoted by the following equation:

$$GVD = \bigcup_{i=1}^{n-1} \bigcup_{j=i+1}^n f(i, j) \tag{5}$$

3. ADJACENT NODE SELECTION ALGORITHM

The details of the adjacent node selection algorithm are shown in Figure 3. Within the search range obtained, we apply IPSO-based path planning algorithm to generate the connection path from the waypoint to all the potential nodes in the range. The connection path is planned in the grid-based map as shown in Figure 3A. The search range in the larger map is shown in Figure 3B, which is also a part of Figure 6. Finally, with the formation of a collision-free path to the adjacent nodes from the two waypoints, we obtain the optimal path selection by calculating the overall path length $\mathcal{L}_c + \mathcal{L}_e$.

To show the necessity of IPSO-based path planning from waypoints to adjacent nodes, a more specific scenario is shown in Figure 4. In Figure 4A, all straight lines connecting nodes and waypoints are separated by obstacles. Among them, points i and point j are located in our search range. If we only rely on the Euclidean distance between node and waypoint, it can be found that the point i is closer to the waypoint. However, after considering the path with obstacle avoidance, the path for point i to the waypoint is longer. To sum up, the obtained search range with ANS algorithm and the IPSO-based path planning algorithm can reduce the computational cost while ensuring a short and safe trajectory. The details of the procedure are described in Algorithm 2.

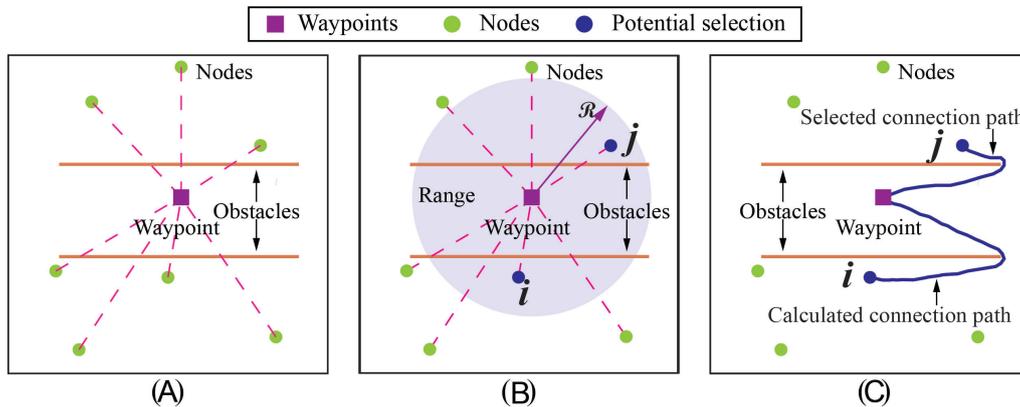


Figure 4. Illustration of ANS method within a more specific sense, where an obstacle obstructing the connection path. (A) The multiple connection paths have been obstructed by the obstacles. (B) It selects the nodes in the defined range. (C) It conducts IPSO point-to-point algorithm to achieve the optimal path to the selected node.

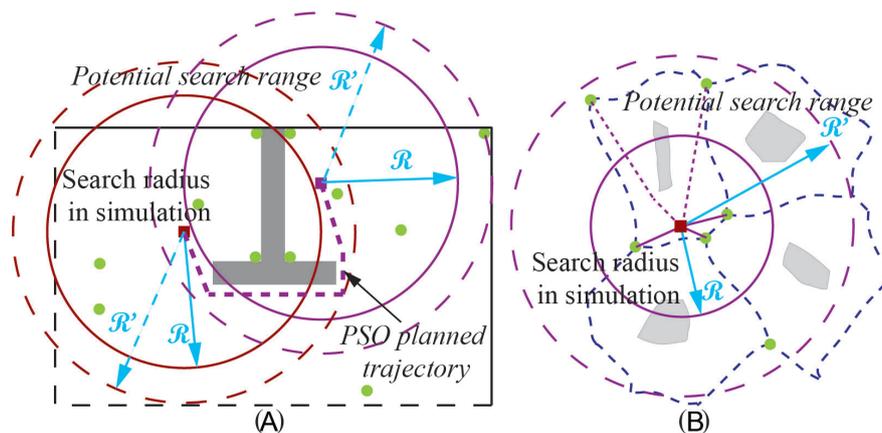


Figure 5. Illustration of the ANS algorithm analysis. (A) From Figure 11 it is enclosed by a pink dashed box. (B) From Figure 13 it is enclosed by a pink dashed box.

We carry out further discussions on our proposed ANS algorithm. Figure 5A and Figure 5B are parts of Figure 11 and Figure 13 (enclosed in pink dashed boxes), respectively. As shown in Figure 5A, the solid circles represent the search radius of the waypoints in the simulation and the red solid dots depict the nodes in the workspace. In Figure 11, we end up with the trajectory that follows the generated safe-aware road. However, if the space in the map is more sparse, our search radius \mathcal{R} may increase to \mathcal{R}' , which may achieve the waypoints in their respective search spaces. Thus, instead of following the safe-awareness road, a new connection path is obtained through the improved PSO algorithm directly, as shown in dashed lines in Figure 5A. Moreover, the sparseness of the overall workspace may not represent the complexity of local obstacles. Therefore, the choice of the radius of the search space may require more mathematical proof and analysis.

With the increasing radius of the potential search range, more nodes are applicable for selection, such as the nodes connected with dashed lines in Figure 5B. Enlarging the search space may avoid some unnecessary detours and give a shorter path. Therefore, the trade-off between path length and safety of the autonomous robot still requires more consideration.

4. IMPROVED PSO-BASED MULTI-WAYPOINT NAVIGATION

The particle swarm optimization (PSO) algorithm is a swarm-based bio-inspired algorithm based on the behavioral observation of birds. It uses an iterative methodology to optimize randomly initialized particles to define a path from the initial position to the goal^[27]. In this section an improved PSO (IPSO) algorithm by introduction of a weighted particles is addressed to resolve the multi-waypoint sequence issue.

4.1. Multi-waypoint visiting sequence

In real-world scenarios, one important factor is that the GPS coordinates provide portions for the multiple waypoints. Traveling from one waypoint to another, the distance between them determines their associated cost. The primary purpose of traveling from one waypoint to another is to simultaneously find the minimal cost of all generated trajectories. Using the coordinates of each waypoint and the PSO algorithm, the minimal-distance path can be found within the environment. The PSO algorithm finds the best waypoint visiting sequence by initializing randomized particles. The algorithm denotes the local best position as x^b and the global best position as x^g . Then by taking advantage of a fitness function, the algorithm guides each particle towards the local and global best positions. The particle velocities are updated as follows:

$$v_p(t+1) = v_p(t) + \alpha_1 \omega_1 [x_p^b(t) - x_p(t)] + \alpha_2 \omega_2 [x_p^g(t) - x_p(t)] \quad (6)$$

$$x_p^{(t+1)} = x_p^{(t)} + v_p^{(t+1)} \quad (7)$$

where, $v_p(t)$ represents the velocity of particle p at instant t , $x_p(t)$ is the position of particle p at instant t , α_1 and α_2 are the positive acceleration constants used to scale the contribution of cognitive and social components. ω_1 and ω_2 are the uniform random number between 0 and 1. $x_p^b(t)$ is the best position the particle p achieved up to instant t at current iteration, $x_p^g(t)$ is the global position that any of p 's neighbors has reached up to instant t . However, if a particle p_i lies close to the $x_p^b(t)$ and $x_p^g(t)$, only one term guides the p_i to search the potential solution. The optimization process in our navigation issue is more likely trapped in local minima. Thus, an improved PSO algorithm is utilized to provide a more promising search direction for all particles during the optimization process.

$$x^w = \sum_{i=1}^P \bar{\alpha}_i^w x_i^b(t) \quad (8)$$

$$\bar{\alpha}_i^w = \frac{\hat{\alpha}_i^w}{\sum_{j=1}^P \hat{\alpha}_j^w} \quad (9)$$

$$\hat{\alpha}_i^w = \frac{\max_{1 \leq k \leq M} (\mathcal{F}(x_k^b(t))) - \mathcal{F}(x_i^b(t)) + \varepsilon}{\max_{1 \leq k \leq M} (\mathcal{F}(x_k^b(t))) - \min_{1 \leq k \leq M} (\mathcal{F}(x_k^b(t))) + \varepsilon}, \quad i = 1, 2, \dots, M, \quad (10)$$

where ε is a positive constant, $\hat{\alpha}$ is the weighted constant of each particle. $\mathcal{F}(\cdot)$ is the fitness function. The worst and the best fitness values of all personal best particles are represented by $\max_{1 \leq k \leq M} (\mathcal{F}(x_k^b(t)))$ and $\min_{1 \leq k \leq M} (\mathcal{F}(x_k^b(t)))$, respectively. The order is optimized through this method, in which each waypoint is visited. A sequence of particles are initialized to compose a population in the original PSO algorithm. A possible optimal solution to an optimization issue in our multi-waypoint sequence is discovered by one particle in

the PSO. This particle indicates a possible optimal solution to the multi-waypoint navigation issue and moves to explore an optimal solution in a certain search space. In this paper, a weighted particle is introduced into a swarm to suggest a more reasonable search direction for all the particles. As a result, the best position of particle and neighbor guides the particle to move along the corrected direction for better convergence.

The multi-waypoint visiting sequence problem can be used to solve the transportation planning problem and Covid-19 disinfection robot path planning in hospitals, in which agents (vehicles) need to be delivered as well as the overall cost and time need to be minimized^[14]. The algorithm of the improved PSO to finding multi-waypoint visiting sequence is explained in Algorithm 1. The objective of the algorithm is to minimize the total trajectory length of Cartesian coordinates (X_n, Y_n) of waypoints given.

In the IPSO model the \mathcal{P}_A represents the particle agent, which is denoted by the $\mathcal{P}_A : \{\mathcal{P}_b, \mathcal{P}_g, \mathcal{N}\}$. \mathcal{N} holds a group of particle agents which are predetermined as neighbors of \mathcal{P}_A . The \mathcal{P}_A are defined by the following parameters.

- (1) Each \mathcal{P}_A requests each neighbor's current personal best location.
- (2) Each \mathcal{P}_A returns its current personal best to neighbors.
- (3) Obtains the center location of each surrounding cluster.
- (4) Determines if the current position has been visited.
- (5) Determines if current position is optimal if not record current position.

Every \mathcal{P}_A obtains a set of neighbors of positions during the initial setup. Utilizing a variety of topologies one can create numerous properties of neighboring particle agents to obtain better performance. Within the proposed model, we assume that each \mathcal{P}_A has a static set of neighbors. Each \mathcal{P}_A keeps track of its local best solution, \mathcal{P}_b , which is where a solution closest to the optimal solution is found in the problem space, while the global best solution is recorded in the parameter \mathcal{P}_g . During each iteration the \mathcal{P}_A evaluates its current position and determines if it needs to perform a fitness evaluation, while simultaneous checking if the termination criteria has been met. If the termination criteria have not been met then it updates its \mathcal{P}_b , obtains the neighbor's \mathcal{P}_b and calculates the \mathcal{P}_g , and marks the current position as visited.

4.2. Safety-aware IPSO multi-waypoint path planning

To ensure the robot safely reaches the waypoints via the planned visiting sequence, the safety-aware road is selected to guide the autonomous robot. Nevertheless, there is a problem with the connection path from the location of the waypoints to safety-aware roads. When we integrate the position information of a waypoint in the workspace, it may be necessary to obtain the collision-free connection path length from the waypoint to all nodes, which is computationally expensive. As shown in Figure 6A, with \mathcal{N} nodes obtained in the workspace, there are \mathcal{N} possible connection paths from the waypoints to the nodes in the workspace. With the initial $\mathcal{N} \times \mathcal{N}$ adjacent distance matrix obtained from GVD graph and the increasing \mathcal{M} waypoints in the workspace, the size of new distance matrix is expanded to $(\mathcal{N} + \mathcal{M}) \times (\mathcal{N} + \mathcal{M})$. Since most of the connection path computations are unnecessary, a new adjacent node selection (ANS) algorithm is proposed to reduce the computational effort by restricting the search space in local regions rather than the entire working region.

The local regions are shown in Figure 6B, and the dark blue nodes, such as α and β nodes in the regions, are potential adjacent nodes to connect. The radius \mathcal{R} of the local area shall be determined by the potential workspace, which determines the number of nodes in the search range. For instance, in an area with clustered obstacles, there are many nodes in the environment; thus, the search radius may be small. However, in an area with sparse obstacles, there are fewer nodes in the environment; thus, the search radius needs to be larger to include all potential nodes.

Since the entire workspace is projected through the GVD graph, we can interpret the sparseness of the entire

Algorithm 1: Improved PSO (IPSO) algorithm for waypoint sequencing**Initialize a population of particles**

Set the size of the swarm to S_p , the maximum number of iteration T_{max} .

for $i=1$ to S_p **do**

 Initialize X_i within the search range of (X_{min}, X_{max}) randomly;

 Initialize V_i within the velocity range of (V_{min}, V_{max}) randomly;

$P_i = X_i$;

end

Evaluate each x_p ;

Identify the best position P_g ;

while (a stop criterion is not satisfied & $t < T_{max}$) **do**

for $p=1$ to S_p **do**

 Update α_1 and α_2 by Equation (8), Equation (9), and Equation (10);

$x^w = \sum_{i=1}^P \bar{\alpha}_i^w x_i^b(t)$

end

 ; // The sum of particle weights

$\bar{\alpha}_i^w = (\hat{\alpha}_i^w / \sum_{j=1}^P \hat{\alpha}_j^w)$; // Average the weights together

$\hat{\alpha}_i^w = \frac{\max_{1 \leq k \leq M} (\mathcal{F}(x_k^b(t))) - \mathcal{F}(x_i^b(t)) + \epsilon}{\max_{1 \leq k \leq M} (\mathcal{F}(x_k^b(t))) - \min_{1 \leq k \leq M} (\mathcal{F}(x_k^b(t))) + \epsilon}$; // Calculate the weights of each particle

$v_p(t+1) = v_p(t) + \alpha_1 \omega_1 [x_p^b(t) - x_p(t)] + \alpha_2 \omega_2 [x_p^g(t) - x_p(t)]$; // Update the fitness value for each particle

$x_p^{t+1} = x_p^t + V_p^{t+1}$; // Add the fitness value to the particle position

$x_p^{b,t+1} = x_p^{b,t}$; // Store the initial local best position

$x_p^{g,t+1} = x_p^{g,t}$

end

 ; // Store the initial global best position

Evaluate $\mathcal{F}(x^{t+1})$;

if $\mathcal{F}(x_p^{b,t+1}) < \mathcal{F}(x_p^{t+1})$ **then**

 | Update $x_p^{g,t+1}$; // Update the new local best position

end

if $\mathcal{F}(x_p^{g,t+1}) < \mathcal{F}(x_p^{b,t+1})$ **then**

 | Update $x_p^{g,t+1}$; // Update the new global best position

end

workspace through the distance of the edge list \mathcal{E} . Nevertheless, some edge distances cannot represent the sparseness of the entire workspace, such as the edges enclosed by the red dotted line in [Figure 6B](#).

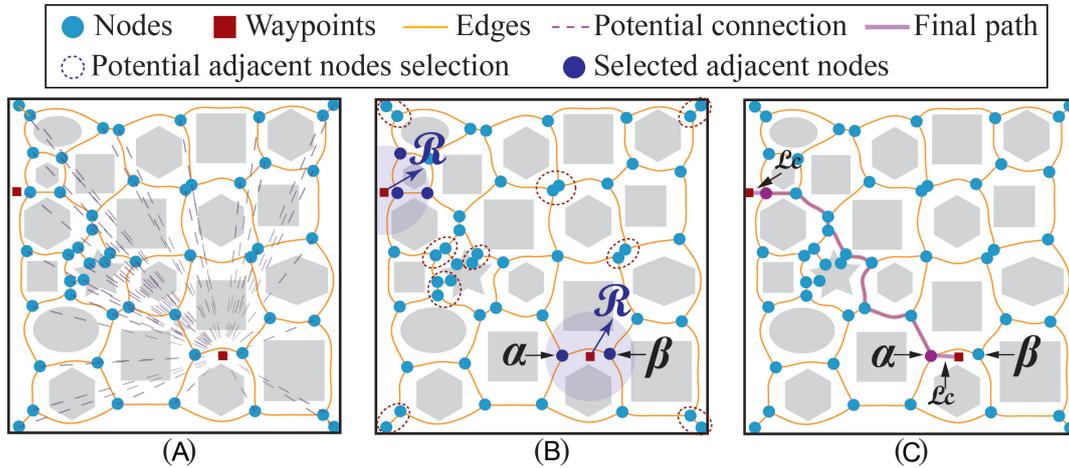


Figure 6. Illustration of the Adjacent Node Selection (ANS) algorithm. (A) The workspace with nodes, edges and waypoints. (B) The node selection in the search range. (C) The final generated path.

Therefore, to exclude these extraneous edges of distance, we sort the entire edge list before removing the lowest 10%. The radius \mathcal{R} of the search range is defined as the average of the remaining edge distance. After obtaining the nodes in the search environment, we plan a collision-free trajectory from the current waypoint to reach each node through the IPSO algorithm.

The IPSO navigation algorithm is initiated in the search range to obtain the optimal path. The local search region is interpreted into grid-based map, where the obstacle areas are inaccessible grids in the workspace. Dijkstra's algorithm is utilized as a local search algorithm and the cost function within the model. For graph-based maps, they must be a method for traveling from one node to another utilizing the edges within the graph. One method of this is Dijkstra's algorithm, which utilizes a weighted graph to determine the shortest path from a source node to a target node. The algorithm also keeps track of the known shortest distances from each node while simultaneously updating their weights to improve the overall shortest path from each node. By recursively establishing a path with random solutions generated in the workspace, the IPSO algorithm can construct the collision-free path with the least fitness value, which also represents the path with minimum length. Therefore, the length of the connecting path \mathcal{L}_c can be obtained, and by combining the length of the GVD path \mathcal{L}_e , the optimal safety-aware trajectory is obtained as shown in Figure 6C.

To effectively reduce and smooth the overall path from each waypoint to the waypoint, various methods have been developed to achieve this goal, for example, the \mathcal{B} -spline curve, which works by taking a set of points that are used to curve sharp close turns around obstacles and is very effective and thus widely used in smooth polylines, due to its closed-form expression of the position coordinates. The original methodology of the \mathcal{B} -spline curve method, in some cases, changes the trajectory of the original path created by the global navigation system. The problem can be mediated by implementing the piecewise \mathcal{B} -spline method, which only smooths the path around each obstacle. Lei *et al.* evaluated the effectiveness of the improved \mathcal{B} -spline method, and ultimately found that the hybrid method was able to reduce the overall all path in point-to-point navigation [38]. The \mathcal{B} -spline curve can be defined by a cardinal functions $L_{j,r}(q)$, control points B_j and degree $(r-1)$, which is given by the following equations.

$$K(q) = \sum L_{j,r}(q)B_j \quad (11)$$

where $B_j = [B_{jx}, B_{jy}]$ are the $(n+1)$ control points and a knot vector u . $N_{i,k}(u)$ are the basic functions, which are defined recursively as follows:

$$L_{j,r}(q) = \frac{(q-x_j)}{x_{j+r-1}-x_j}N_{j,r-1}(q) + \frac{(x_{j+r}-q)}{x_{j+r}-x_{j+1}}N_{j+1,r-1}(q) \quad (12)$$

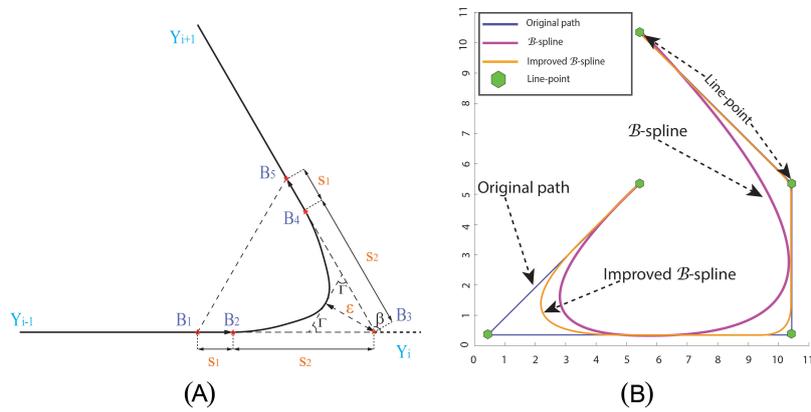


Figure 7. Illustration of the \mathcal{B} -spline function. (A) The improved segmented \mathcal{B} -spline curve. (B) The same path smoothed by the fundamental \mathcal{B} -spline and the improved \mathcal{B} -spline function.

$$L_{j,r}(q) = \begin{cases} 1, & q_j q_j \leq q \leq q_{j+1} \\ 0, & \text{otherwise} \end{cases} ; q \in [0, 1] \quad (13)$$

Geometric continuity G^2 is the metric used to evaluate smoothing methods, which is defined by the tangent unit and curvature vector at the intersection of two continuous segments^[38] [Figure 7A]. To achieve G^2 continuity, the control points B_j of \mathcal{B} -spline curve to the path point, Y_j , is defined as

$$\begin{aligned} B_1 &= Y_j - (1 + v)s_2q_{j-1} \\ B_2 &= -s_2q_{j-1} \\ B_3 &= Y_j \\ B_4 &= Y_j + s_2q_j \\ B_5 &= Y_j + (1 + v)s_2q_{j-1} \end{aligned} \quad (14)$$

where c is smoothing length ratio $v = s_1/s_2$, and q_{j-1} defines the unit vector of $Y_{j-1}Y_j$. q_j represents the unit vector of the line Y_jY_{j+1} . The combined sum of s_1 and s_2 is the smoothed length. The half of the corner angle is denoted as: $\Gamma = \beta/2$. Using a knot vector of $[0, 0, 0, 0, 0.5, 1, 1, 1, 1]$, the smoothing error distance ϵ and maximum curvature K_{max} within the smooth path can be expressed as:

$$\epsilon = \frac{s_2 \sin \Gamma}{2} \quad (15)$$

$$R_{max} = \frac{4 \sin \Gamma}{3s_2 \cos^2 \Gamma} \quad (16)$$

Using the previous equations the smoothing error distance ϵ can be defined by the existing maximum curvature R_{max} given by the robot:

$$\epsilon = \frac{2 \tan^2 \Gamma}{3R_{max}} \quad (17)$$

The improved \mathcal{B} -spline model has specific advantages over the basic \mathcal{B} -spline model, one of which is its ability to smooth many different trajectories with various angles, as seen in Figure 7B. The curve produced by the improved \mathcal{B} -spline model is significantly closer to the original path than the original model. When considering the constraints of the robot, the improved \mathcal{B} -spline mode performs better in various degrees of angles. The overall advantages of the improved \mathcal{B} -spline model are as follows:

- (1) The path generated is tangent and curvature continuity, so that the robot can have a smooth steering command, which can correct any discontinuity of normal acceleration and establish a safer path for the robot to follow.

(2) The improved model generates a better curve by solely affecting the two lines within the corner of the original trajectory. Each curve generated affects others within the lines.

(3) The improved model easily adjusts to the smoothed path based on the environment constraints or the robot.

Algorithm 2: Pseudocode for the adjacent node selection (ANS) algorithm

Input: Edge list \mathcal{E} , \mathcal{N} nodes coordinates $(\mathbb{N}_x, \mathbb{N}_y)$, $\mathcal{N} \times \mathcal{N}$ distance matrix \mathcal{D} and the location of the waypoint α , (α_x, α_y) and the waypoint β , (β_x, β_y) .

Output: The path length of the trajectory \mathcal{L}_t

$N_e = \text{size}(\mathcal{E})$; // Number of the edges in the workspace

$[\mathcal{E}_s, \text{sortInd}] = \text{sort}(\mathcal{E})$; // Sort the edge list from low to high

$N_s = \lceil \frac{N_e}{10} \rceil$; // Exclude 10% of extraneous edge distance

$\mathcal{E}_t = 0$;

for $i = N_s : N_e$ **do**

 | $\mathcal{E}_t = \mathcal{E}_t + \mathcal{E}_s(i)$;

end

$\mathcal{R} = \frac{\mathcal{E}_t}{N_e - N_s}$; // The radius of the search range

for $j = 1 : \mathcal{N}$ **do**

 | **if** $(\mathbb{N}_{j,x}, \mathbb{N}_{j,y})$ in range of \mathcal{R} from α_x, α_y **then**

 | $\mathcal{A}_\alpha = [\mathcal{A}_\alpha; \mathbb{N}_{j,x}, \mathbb{N}_{j,y}]$; // Add the potential adjacent nodes of
 | waypoint α in the list

 | **end**

 | **if** $(\mathbb{N}_{j,x}, \mathbb{N}_{j,y})$ in range of \mathcal{R} from β_x, β_y **then**

 | $\mathcal{A}_\beta = [\mathcal{A}_\beta; \mathbb{N}_{j,x}, \mathbb{N}_{j,y}]$; // Add the potential adjacent nodes of
 | waypoint β in the list

 | **end**

end

$N_\alpha = \text{size}(\mathcal{A}_\alpha)$; $N_\beta = \text{size}(\mathcal{A}_\beta)$; // Number of the points in the search range

Apply IPSO-based path planning algorithm;

Obtain the length of the connection path \mathcal{L}_α and \mathcal{L}_β in the search range;

$\mathcal{L}_t = \infty$;

for $a = 1 : N_\alpha$ **do**

 | **for** $b = 1 : N_\beta$ **do**

 | $\mathcal{L}_{temp} = \mathcal{L}_\alpha(a) + \mathcal{D}(\mathbb{N}_{\alpha,a}, \mathbb{N}_{\beta,b}) + \mathcal{L}_\beta(b)$; // Calculate the total path

 | **end**

 | **if** $\mathcal{L}_t > \mathcal{L}_{temp}$ **then**

 | $\mathcal{L}_t = \mathcal{L}_{temp}$; // Achieve the minimum path length

 | **end**

end

5. REACTIVE LOCAL NAVIGATION

A crucial aspect in developing a multi-waypoint model is accounting for moving and unknown obstacles [39,40]. In a real-world setting, not all objects are static and known. To develop a more efficient model, we propose the use of a local navigator to remedy this issue.

In order to avoid dynamic and unknown obstacles, the proposed model employs the Vector Field Histogram

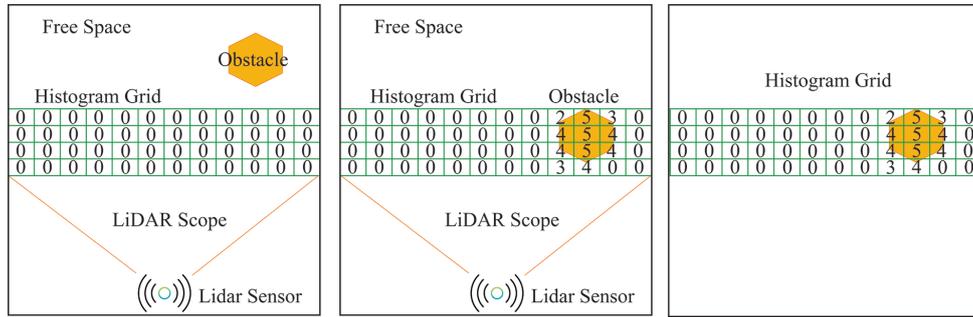


Figure 8. Illustration of how the VFH uses a probability along with histogram-based grid to detect and build a map simultaneously.

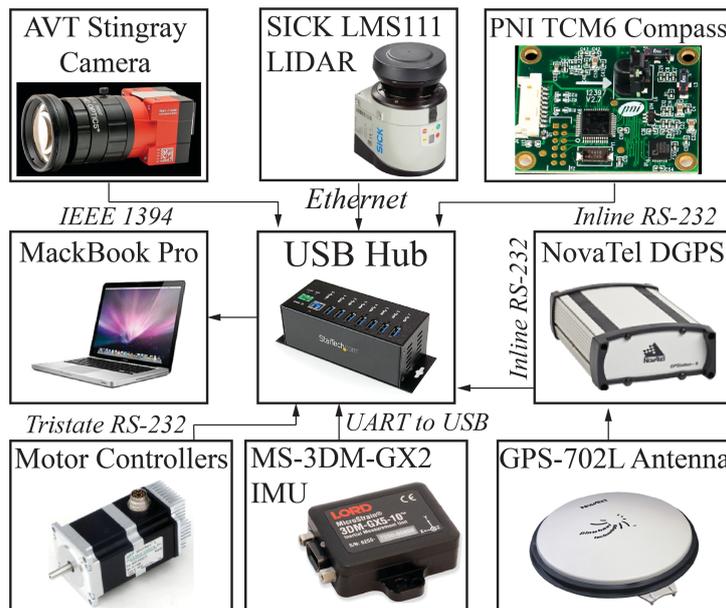


Figure 9. Robot sensor configuration for multi-waypoint navigation and mapping.

(VFH) model as a reactive local navigator. An autonomous robot uses a velocity command to and from each waypoint, provided by the local navigator^[41]. By applying the VFH to the overall global trajectory with a sequence of markers, the path can be broken down into various segments to improve the efficiency in obstacle-populated workspaces. The local navigator builds a map depicting the free space and obstacles in the map by utilizing a 2D histogram grid with equally sized cells^[42]. As the robot follows the generated trajectory within the workspace, the map is simultaneously built, shown in [Figure 8](#). In developing an autonomous obstacle avoidance model, concurrent map building and navigation are crucial. The robot pose (X, Y, Yaw) is used to determine the map building. Thus, the precise registration of the built local map as a part of the global map can be carried out. This map building aims to construct an occupancy-cell-based map. The values for each cell in the map vary over the range $[-127, 128]$ ^[42]. The initial value is zero, which indicates that the cell is neither occupied nor unoccupied. The value is 128 if one cell is occupied with certainty and -127 if one cell is unoccupied with certainty. The values falling into $(-127, 128)$ express contain a level of certainty in the range. When the VFH model is employed in conjunction with the GVD and IPSO algorithm, the robot can be successfully navigated through our built map with obstacle avoidance. In combination with our local navigator, a sensor configuration can be developed for the local navigator to perform it. In [Figure 9](#) one can see the overview of our sensor configuration. The proposed configuration utilizes a 270-degree SICK LMS LiDAR sensor to detect obstacles within a range of 20 m @ 0.25-degree resolution. The LiDAR sensor scans at a rate of 25Hz. Then, it needs a method of finding our current position and the waypoints within the map. A

Table 1. Comparison of minimum path length, average path length, STD of path length, minimum time, average time and STD of time with other models. The parameter for the test of each model was: 100 initialized particles, 10 runs per data set, and a maximum of 10 minutes per run

Datasets	Model	Min length (m)	Average length (m)	Length STD (m)	Min time (s)	Average time (s)	Time STD (s)
Ch150	Proposed model	1.67E+04	1.77E+04	5.99E+02	1.75E+04	1.31E+01	6.09E-02
	ACO	1.84E+04	2.23E+04	4.40E+03	1.67E+04	1.76E+02	1.32E+00
	GA	4.22E+04	5.04E+04	5.38E+03	1.79E+04	1.39E-02	1.25E-03
	SA	2.47E+04	3.05E+04	4.11E+03	1.75E+04	6.51E+00	7.25E-01
	GWO	3.23E+04	3.66E+04	2.64E+03	5.78E-01	9.39E-01	1.73E-01
	SOM	4.26E+04	4.70E+04	2.932E+03	1.84E+03	2.26E+03	1.77E+02
	ICA	3.29E+04	3.50E+04	9.34E+02	1.55E+03	2.04E+03	1.79E+02
	Proposed model	1.09E+05	1.17E+05	5.50E+03	2.15E+01	2.15E+01	9.73E-02
KroA200	ACO	1.30E+05	2.81E+05	4.09E+05	2.44E+02	5.17E+02	8.36E+02
	GA	2.39E+05	2.57E+05	1.15E+04	1.21E-02	1.54E-02	4.45E-03
	SA	1.96E+05	2.13E+05	1.18E+04	6.27E+00	6.62E+00	8.70E-01
	GWO	2.17E+05	2.40E+05	1.24E+04	1.12E+00	1.45E+00	3.68E-01
	SOM	2.13E+05	2.60E+05	2.192E+04	5.49E+03	2.60E+05	2.31E+04
	ICA	2.12E+05	2.60E+05	6.671E+03	3.22E+02	6.81E+02	1.38E+02
	Proposed model	2.77E+05	2.89E+05	5.92E+03	4.28E+01	4.29E+01	6.06E-02
	ACO	3.37E+05	4.33E+05	4.82E+04	2.09E+02	2.37E+02	1.41E+01
PR299	GA	3.19E+05	3.44E+05	1.17E+04	3.24E-02	3.38E-02	8.43E-04
	SA	5.26E+05	5.59E+05	1.94E+04	6.27E+00	6.52E+00	6.68E-01
	GWO	2.90E+05	3.90E+05	8.15E+04	6.71E+01	8.00E+01	7.51E+00
	SOM	2.72E+05	3.15E+05	6.34E+04	4.25E+03	4.75E+03	3.45E+02
	ICA	4.81E+05	4.96E+05	1.34E+04	2.70E+03	2.81E+03	7.60E+01
	Proposed model	1.11E+05	1.14E+05	1.60E+03	1.41E+02	1.42E+02	3.65E-01
	ACO	—	—	—	—	—	—
	GA	1.37E+05	1.93E+05	6.30E+04	9.36E-02	9.57E-02	2.19E-03
PA561	SA	1.88E+05	1.92E+05	2.07E+03	6.27E+00	6.42E+00	4.29E-01
	GWO	3.02E+05	4.21E+05	5.94E+04	7.64E+01	8.26E+01	2.21E+00
	SOM	1.01E+05	1.21E+05	1.84E+04	1.1E+01	8.19E+02	3.94E+02
	ICA	1.48E+05	1.51E+05	2.43E+03	1.22E+03	2.04E+03	1.79E+02
	Proposed model	—	—	—	—	—	—

Novatel's ProPak-LB Plus DGPS sensor is utilized to obtain our current position and how it correlates to the coordinates of each waypoint. Next, a PNI TCM6 digital compass is employed to establish our heading with an accuracy of 0.5 degrees. The sensor updates at 20Hz, which allows the robot to operate efficiently. Lastly, the configuration utilizes an AVT Stingray F-080C 1/3" CCD camera, which enables our robot to sense obstacles of various heights, shapes and sizes. The stingray camera is perfect for robot vision because it uses the IIDC IEEE 1394B protocol to transfer images. The system needs a computer system to house our operating system, sensor data, and programs for the robot. In this portion of the sensor configuration, a MacBook Pro is equipped to suit our needs. The last step in the process is to establish a method of communication from the sensors to the computer systems. A sort of UART to USB hub is utilized for this purpose and fuses the sensor data together without losing any sensor information. The type of sensor confusion can be used on most ground-based robot systems for indoor and outdoor use.

6. SIMULATION AND COMPARISON STUDIES

In this section, simulation and comparison studies are performed to illustrate the value and vitality of the proposed model. In the first experiment, simulations are conducted using a well-known Traveling Salesman Problem (TSP) based data set, and results are compared with other heuristic-based algorithms. The proposed model is thoroughly evaluated in the second experiment through a comparison study using a similar model proven to work effectively for multi-waypoint navigation.

6.1. Comparison studies with benchmark datasets

To show the effectiveness of our IPSO waypoint sequencing model, a comparison study was conducted with well-known TSP data sets and various heuristic-based algorithms. The employed datasets and algorithms are as follows: (a) 561-city problem by Kleinschmidt (pa561); (b) 299-city problem by Patberg/Rinaldi (pr299); (c) 200-city problem A, by Krolik/Felts/Nelson (kroA200); and (d) 150-city problem by Chur Ritz (ch150). The selected datasets have been verified and widely used to prove the validity of multi-waypoint sequencing

Table 2. An illustration of the number of nodes, distance, and time spent traversing with the map to each waypoint

Model	Nodes	Distance	Time spent (s)
Zhang's model before node reduction	242	271.1	2.25
Zhang's model after node reduction	24	253.4	0.66
Proposed model	38	277.7	0.40

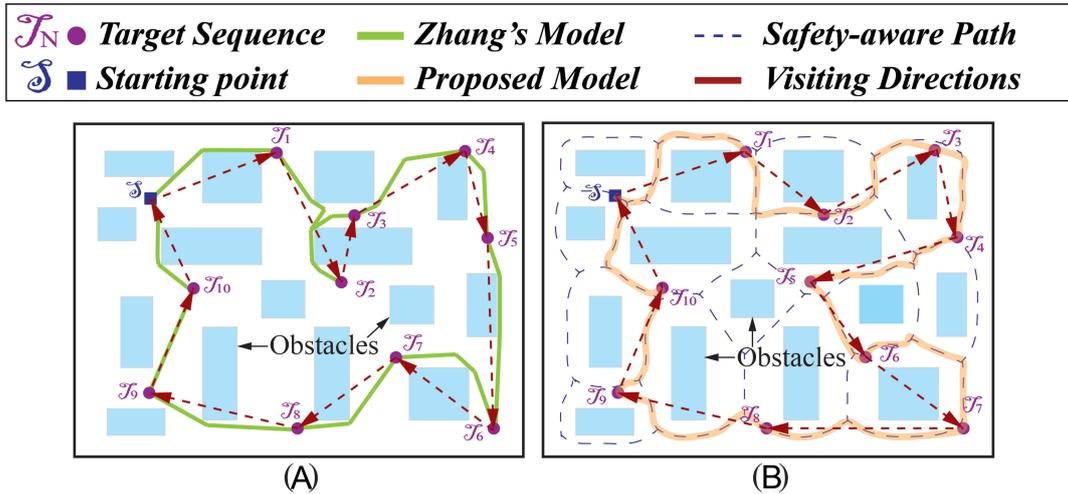


Figure 10. Illustration of the path created from the other models [43]. (A) It depicts the path created by Zhang et al.'s model by the green lines (redrawn by Zhang et al., 2021 [43]). (B) It represents the proposed method point order and traversed path. The point order is illustrated by the violet arrows, while the orange path represents the robot path. Safety-aware roads are depicted by the blue dashed lines. The waypoints are illustrated by the violet circles.

models. The Simulated Annealing (SA) algorithm, Grey Wolf Optimization (GWO) algorithm, Ant Colony Optimization (ACO) algorithm, Genetic Algorithms (GA), Imperialist Competitive Algorithm (ICA), and Self-Organizing Maps (SOM) were chosen as the heuristic-based algorithms used in the comparison studies. The ICA algorithm is a biologically inspired algorithm by the human, which simulates the social-political process of imperialism and imperialistic competition. The SOM algorithm is similar to a typical artificial neural network algorithm, except it utilizes a competitive learning process instead of backpropagation that utilizes gradient descent.

Heuristic-based algorithms have similar attributes; due to this feature, the same parameters can be used to construct a stable comparison study for our proposed IPSO algorithm. The conducted comparison studies focus on six key attributes such as: min length (m), average length (m), length standard deviation (m), min time (s), average time (s), and time standard deviation (s). The variance between each algorithm can be seen by assessing each parameter. The above analyses show how effective the IPSO model can generate the minimum overall global trajectory in Table 1. The global trajectory generated by the compared algorithms is notably larger than the IPSO model. However, regarding the time aspect, the IPSO model was unable to achieve the shortest time. The significance of the proposed model can be seen in the STD evaluation parameter. The results of the comparison studies more than show the validity and performance of the proposed model to discover the optimal waypoint visiting sequence.

6.2. Model comparison studies

The compared models were developed to address the issues of multi-waypoint navigation and mapping in various applications. Each model uses some variation of a global navigation system in combination with an obstacle avoidance technique. The models were selected based on their map configuration and overall efficiency in solving the multi-waypoint navigation problem. Our comparison studies analyze the number of nodes, the trajectories produced, and the total time to fulfill the fastest route.

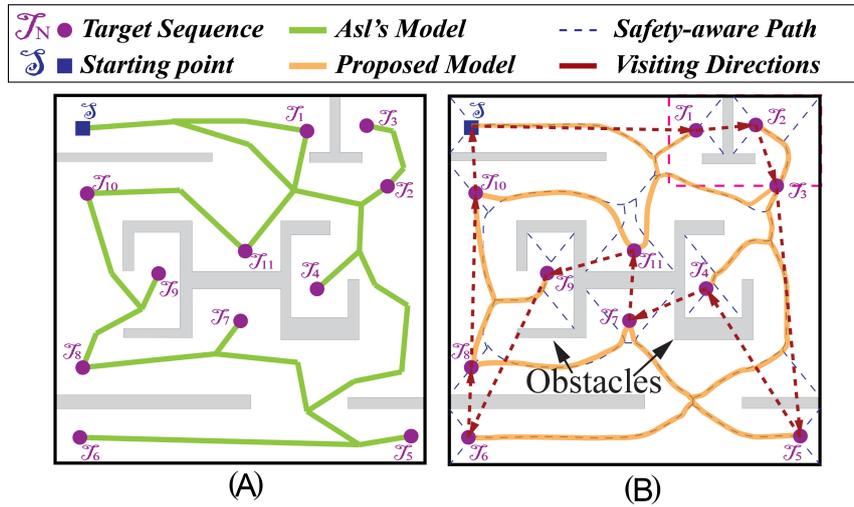


Figure 11. Illustration of the path created from the other models^[44]. (A) It depicts the path created by Asl and Taghirad model by the green lines (redrawn by Asl and Taghirad, 2019^[44]). (B) It represents the proposed method point order and traversed path. The point order is illustrated by the violet arrows, while the orange path represents the robot path. Safety-aware roads are depicted by the blue dashed lines. The waypoints are illustrated by the violet circles.

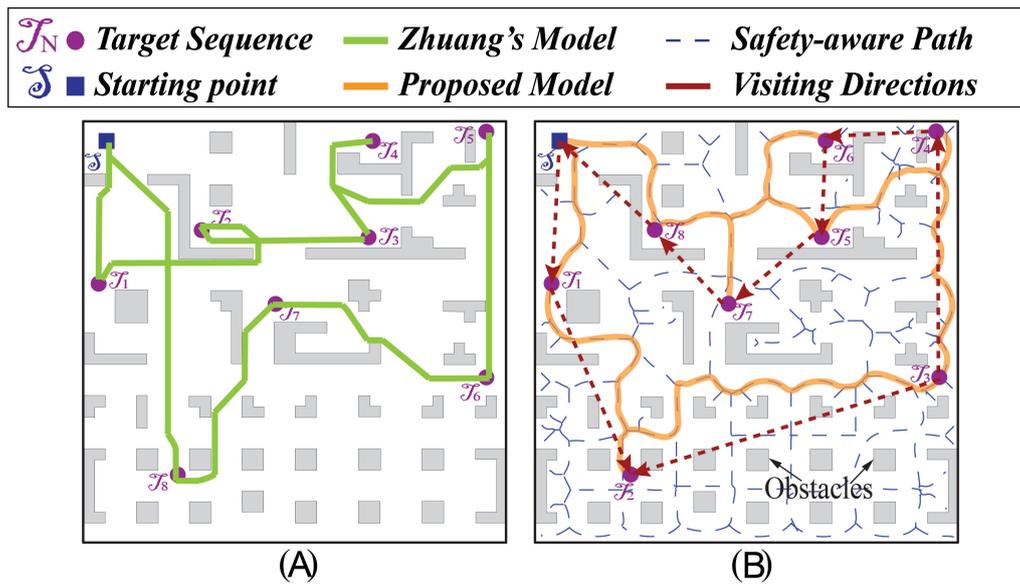


Figure 12. Illustration of the path created from the compared models^[45]. (A) It depicts the path created by Zhuang et al.'s model by the green lines (redrawn from Zhuang et al., 2021^[45]). (B) It represents the proposed method point order and traversed path. The point order is illustrated by the violet arrows, while the orange path represents the robot path. Safety-aware roads are depicted by the blue dashed lines. The waypoints are illustrated by the violet circles.

It is clear that the waypoint order and paths obtained by each model are created in an obstacle-free environment, as illustrated in Figure 10A. The length created by the Zhang's model was 240.84 m, while the proposed model produced a shorter trajectory of 219.99 m. This is due to the founded waypoint orders in the environment. In Zhang's comparison study, the proposed model establishes more nodes, and the overall path is expanded by 1.09%, but the proposed model generates a solution 6.1% faster than the compared model. Zhang's proposed model has to utilize a node selection algorithm to establish its shortest path, while the proposed model does not. Due to this feature, the compared model was evaluated before this crucial step and discovered that the nodes established were vastly greater than the proposed model, as seen in Table 2. Considering this factor, the proposed model can surpass and outperform Zhang's model. Asl and Taghirad aimed to solve the multi-goal

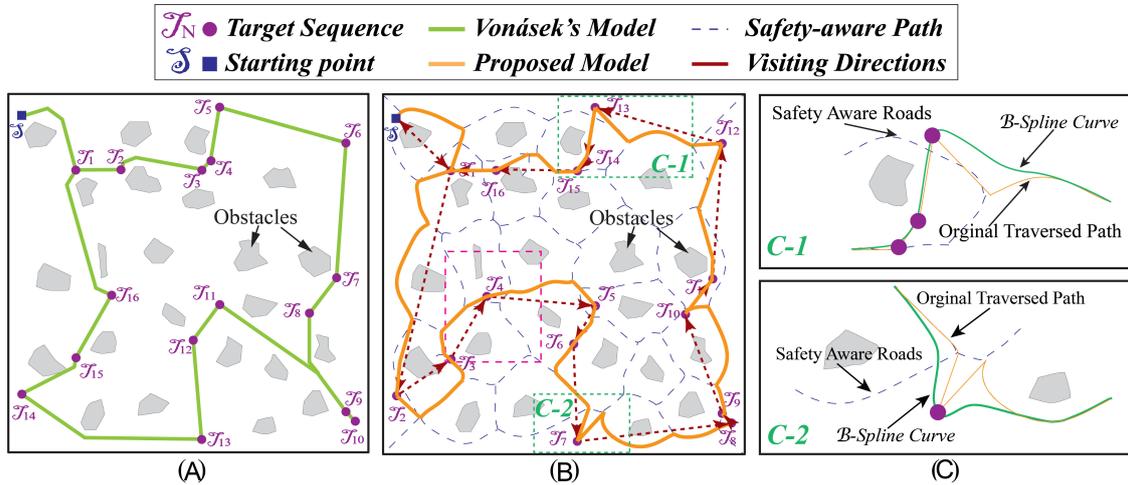


Figure 13. Illustration of the path created from the compared models [46]. (A) It depicts the path created by Vonásek’s model shown by the green lines (redrawn from Vonásek and Penicka, 2019 [46]). (B) It represents the proposed method point order and traversed path. The point order is illustrated by the violet arrows, while the orange path represents the robot path. Safety-aware roads are depicted by the blue dashed lines. The waypoints are illustrated by the violet circles. (C) It depicts how the B -spline curve is applied to the known path, which smooths and reduces the path for a local navigator to traverse.

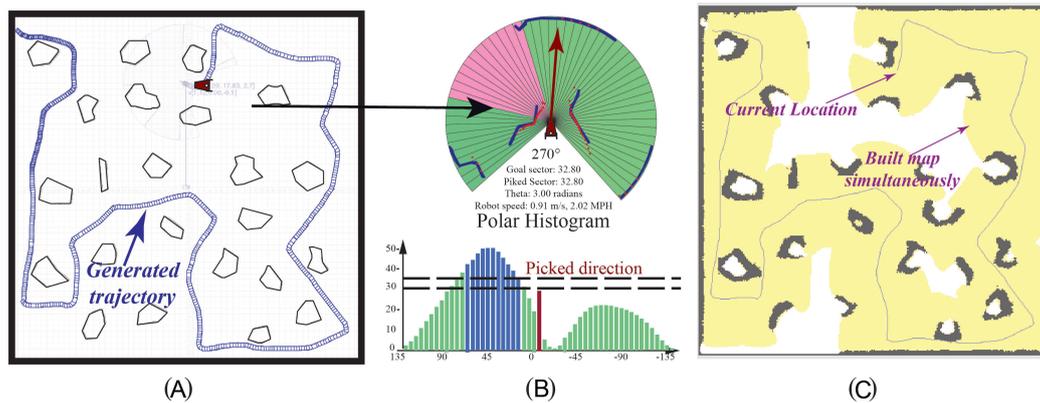


Figure 14. Illustration of the scenario in Figure 13 navigation and mapping simulation. (A) It depicts the robot traversing a majority of the map, while avoiding obstacles in the environment. (B) It illustrates a polar histogram and how the obstacles are viewed while the robot is in motion, as well as points of high impact. The obstacles are viewed as lines since the LiDAR sensor can only see the part of the object that faces the LiDAR sensor. The picked direction portion of part (B) depicts the probability of colliding with obstacles while also selecting the best direction to move the robot. (C) It demonstrates the map being simultaneously built as the the robot traverses the established trajectory.

navigation problem by developing a traveling salesman problem in the belief space [44].

From Figure 11, one can see the established path using both Asl’s method as well as the method proposed in this paper. The method proposed by Asl and Taghirad has the advantage of creating a shorter path but requires a greater number of nodes than the proposed method. Figure 12 depicts the model comparison between Zhuang *et al.*’s model and the proposed method [45]. The simulation studies reveal that the proposed model had an increased length of approximately 0.05% over Zhuang *et al.*’s model. Once the compared model requires a greater number of nodes to complete its multi-waypoint navigation, another key point from this comparison is the path created from Zhuang *et al.*’s model and the proximity to the obstacles in the map. In a real-world environment, the robot could obtain server damage or cause an accident if it is too close to the surrounding obstacles [45]. The proposed method established an effective path without risking the robots well being. Vonásek and Penicka [46] models had similar results to the previous models, with an increased length

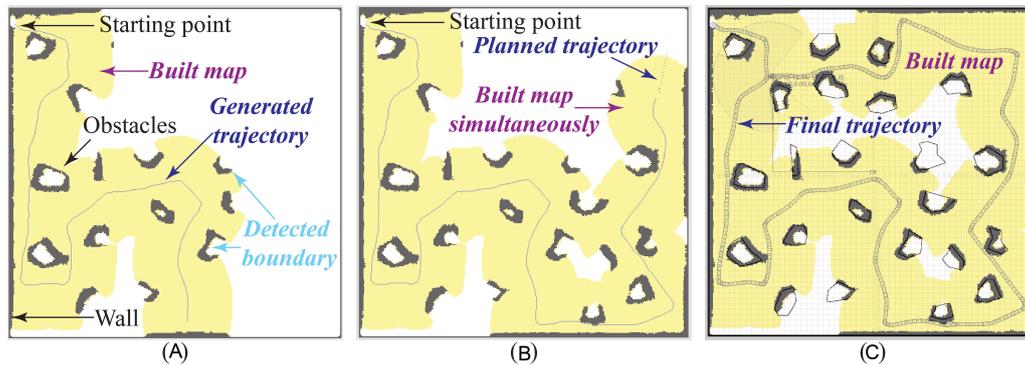


Figure 15. Illustration of the navigation and mapping ability of the proposed model. (A) The robot follows the generated trajectory and detects obstacle boundaries by the LiDAR sensor. (B) The simultaneous map building and navigation capabilities of the proposed model. (C) The fully generated trajectory and established map.

of approximately 0.05%, as seen in Figure 13. The two compared models have the same problem as Zhuang *et al.*' model [45]. The paths are excessively close to obstacles in the map and thus are not efficient for real-world implementation.

In such an environment, it is very important to consider the robot safety because of the narrow paths created being tightly packed with triangular shaped obstacles. Although most of our model comparison results showed that the path constructed with the proposed model increased from the compared models, we achieved our goal of constructing safety-aware roads for robot safety and establishing an obstacle-free path.

From Figure 14, it is observed how the local navigator establishes a map through vision sensors such as LiDAR. In Figure 14, it is obvious that the map in various stages is shown as the robot traverses along the generated trajectory found in the Vonásek's simulation [Figure 13] [46].

The robot is able to reconstruct the outer boundary of the obstacles through the LiDAR scan. These are depicted as the poly-shaped figures with a rough background and a white center.

In Figure 15A, it is clear to see the original starting position as well as the planned trajectory, which was found utilizing our proposed IPSO model. From the figures, one could observe fully and partly detected obstacles as well as the outer boundary being detected. The map depicted in Figure 15 has a height and width of 60 m. The dimensions of the robot are approximately 0.82 m long and 0.68 m wide. In Figure 15A, the robot has successfully traversed one third of the map, while simultaneously avoiding the obstacles. In Figure 15B, it illustrates the robot's planned trajectory and the map being simultaneously constructed along the path. The portions of the figure depicted in a yellow field are the built map sensed by the onboard sensors. Finally, in Figure 15C, the robot has successfully visited each waypoint and reached its final destination, and it shows the complete depiction of the map along the projected path.

7. CONCLUSION

We proposed an adjacent node selection (ANS) algorithm to find a node in the graph to connect waypoints. This algorithm is utilized in the safety-aware multi-waypoint navigation and mapping by an improved PSO and GVD model. An IPSO-based multi-waypoint algorithm has been developed to define an order for waypoint navigation. Through our proposed ANS algorithm, connections among the waypoints and the safety-aware routes to reach multi-objective optimization can be created. The feasibility and effectiveness of our model by conducting a benchmark test and model comparison studies and analyses have been demonstrated.

DECLARATIONS

Acknowledgments

The authors would like to thank the editor-in-chief, the associate editor, and the anonymous reviewers for their valuable comments.

Authors' contributions

Made substantial contributions to the research, idea generation, algorithm design, simulation, wrote and edited the original draft: Sellers T, Lei T, Luo C

Performed critical review, commentary and revision, as well as provided administrative, technical, and material support: Jan G, Ma J

Financial support and sponsorship

None.

Availability of data and materials

Not applicable.

Conflicts of interest

All authors declared that there are no conflicts of interest.

Ethical approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Copyright

© The Author(s) 2022.

REFERENCES

1. Chu Z, Wang F, Lei T, Luo C. Path planning based on deep reinforcement learning for autonomous underwater vehicles under ocean current disturbance. *IEEE Trans Intell Veh* 2022;1–1. [DOI](#)
2. Zhao W, Lun R, Gordon C, et al. A privacy-aware Kinect-based system for healthcare professionals. In: IEEE International Conference on Electro Information Technology. Grand Forks, USA; 2016. pp. 205–10. [DOI](#)
3. Zhao W, Lun R, Gordon C, et al. Liftingdoneright: a privacy-aware human motion tracking system for healthcare professionals. *Int J Handheld Comput Res* 2016;7:1–15. [DOI](#)
4. Lei T, Luo C, Jan GE, Bi Z. Deep learning-based complete coverage path planning with re-joint and obstacle fusion paradigm. *Front Robot AI* 2022;9. [DOI](#)
5. Li X, Luo C, Xu Y, Li P. A Fuzzy PID controller applied in AGV control system. In: International Conference on Advanced Robotics and Mechatronics. Macau, China; 2016. pp. 555–60. [DOI](#)
6. Lei T, Luo C, Ball JE, Bi Z. A hybrid fireworks algorithm to navigation and mapping. In: Handbook of Research on Fireworks Algorithms and Swarm Intelligence. IGI Global; 2020. pp. 213–32. [DOI](#)
7. Jayaraman E, Lei T, Rahimi S, Cheng S, Luo C. Immune system algorithms to environmental exploration of robot navigation and mapping. In: International Conference on Swarm Intelligence. Qindao, China: Springer; 2021. pp. 73–84. [DOI](#)
8. Yang J, Chai T, Luo C, Yu W. Intelligent demand forecasting of smelting process using data-driven and mechanism model. *IEEE Trans Ind Electron* 2018;66:9745–55. [DOI](#)
9. Zhu D, Tian C, Jiang X, Luo C. Multi-AUVs cooperative complete coverage path planning based on GBNN algorithm. In: Chinese Control and Decision Conference. Chongqing, China; 2017. pp. 6761–66. [DOI](#)
10. Luo C, Yang SX, Mo H, Li X. Safety aware robot coverage motion planning with virtual-obstacle-based navigation. In: IEEE International Conference on Information and Automation. Lijiang, China; 2015. pp. 2110–15. [DOI](#)
11. Wang L, Luo C. A hybrid genetic tabu search algorithm for mobile robot to solve AS/RS path planning. *Int J Rob Autom* 2018;33:161–68. [DOI](#)
12. Wang L, Luo C, Cai J. A variable interval rescheduling strategy for dynamic flexible job shop scheduling problem by improved genetic algorithm. *J Adv Transp* 2017;2017. [DOI](#)

13. Qu Q, Zhang H, Luo C, Yu R. Robust control design for multi-player nonlinear systems with input disturbances via adaptive dynamic programming. *Neurocomputing* 2019;334:1–10. DOI
14. Penicka R, Scaramuzza D. Minimum-time quadrotor waypoint flight in cluttered environments. *IEEE Rob Autom Lett* 2022;7:5719–26. DOI
15. Jan GE, Luo C, Lin HT, Fung K. Complete area coverage path-planning with arbitrary shape obstacles. *J Autom Contr Eng* 2019;7. Available from: https://dl.wqtxts1x7le7.cloudfront.net/79144324/20191218032728291-with-cover-page-v2.pdf?Expires=1661938236&Signature=F9H9SowZfl7P9v74Xs1quZpkYuja0eBkKJx1C--LkgBppman~V~hvtZqk1Z-ChAOsmQM1p2471dlhAmBjm8EsnfvOqvVrjflCVHUK-opgkdCHuQgdFors7oKBCBkgHI0tg9tVHAND8hgv0izfLx3YKuQeNidjvTlfXT66CbSDDtsfuZUe6wTnKYrVIq993iBCwGpQmWZ8nhQGPYqiw-fC06aBPWusmC9jCpidVZpb6R0Iv1Z-ii7gnj1~rdOG6Aga1qUISEPR7csU~9-oPDnZ2FZbXqvc1uge75e4JhqmG3sMdzLlFosJofxK-3FflNLakRVxpWYA20EbTviA__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA [Last accessed on 31 Aug 2022].
16. Lei T, Sellers T, Rahimi S, Cheng S, Luo C. A nature-inspired algorithm to adaptively safe navigation of a Covid-19 disinfection robot. In: International Conference on Intelligent Robotics and Applications. Cham: Springer; 2021. pp. 123–34. DOI
17. Lei T, Luo C, Sellers T, Wang Y, Liu L. Multi-task allocation framework with spatial dislocation collision avoidance for multiple aerial robots. *IEEE Trans Aerosp Electron Syst* 2022. DOI
18. Chen Y, Liu C, Shi BE, Liu M. Robot navigation in crowds by graph convolutional networks with attention learned from human gaze. *IEEE Rob Autom Lett* 2020;5:2754–61. DOI
19. Chen J, Luo C, Krishnan M, Paulik M, Tang Y. An enhanced dynamic Delaunay triangulation-based path planning algorithm for autonomous mobile robot navigation. In: Intelligent Robots and Computer Vision XXVII: Algorithms and Techniques. vol. 7539. California, USA: SPIE; 2010. pp. 253–64. DOI
20. Luo C, Xiao Y, Yang SX, Jan GE. Improving vehicle navigation by a heading-enabled ACO approach. In: *World Automation Congress*. Rio Grande, USA: IEEE; 2016. pp. 1–6. DOI
21. Lei T, Luo C, Jan GE, Fung K. Variable speed robot navigation by an ACO approach. In: *International Conference on Swarm Intelligence*. Cham: Springer; 2019. pp. 232–42. DOI
22. Wang L, Luo C, Li M, Cai J. Trajectory planning of an autonomous mobile robot by evolving ant colony system. *Int J Rob Autom* 2017;32:406–13. DOI
23. Lei T, Luo C, Sellers T, Rahimi S. A bat-pigeon algorithm to crack detection-enabled autonomous vehicle navigation and mapping. *Intell Syst Appl* 2021;12:200053. DOI
24. Luo C, Yang SX, Krishnan M, Paulik M. An effective vector-driven biologically-motivated neural network algorithm to real-time autonomous robot navigation. In: IEEE International Conference on Robotics and Automation. Hong Kong, China; 2014. pp. 4094–99. DOI
25. Na YK, Oh SY. Hybrid control for autonomous mobile robot navigation using neural network based behavior modules and environment classification. *Auton* 2003;15:193–206. DOI
26. Zhong C, Luo C, Chu Z, Gan W. A continuous hopfield neural network based on dynamic step for the traveling salesman problem. In: International Joint Conference on Neural Network. Anchorage, USA: IEEE; 2017. pp. 3318–23. DOI
27. Lazreg M, Benamrane N. Hybrid system for optimizing the robot mobile navigation using ANFIS and PSO. *Rob Auton Syst* 2022;153:104114. DOI
28. Jensen-Nau KR, Hermans T, Leang KK. Near-optimal area-coverage path planning of energy-constrained aerial robots with application in autonomous environmental monitoring. *IEEE Trans Autom Sci Eng* 2021;18:1453–68. DOI
29. Janoš J, Vonásek V, Pěnička R. Multi-goal path planning using multiple random trees. *IEEE Rob Autom Lett* 2021;6:4201–8. DOI
30. Ortiz S, Yu W. Autonomous navigation in unknown environment using sliding mode SLAM and genetic algorithm. *Intell Robot* ;1:131–50. DOI
31. Bernardo R, Sousa J, Gonçalves PJ. Planning robotic agent actions using semantic knowledge for a home environment. *Intell Robot* 2021;1:101–15. DOI
32. Luo C, Yang SX. A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments. *IEEE Trans Neural Netw* 2008;19:1279–98. DOI
33. Shair S, Chandler JH, González-Villela VJ, Parkin RM, Jackson MR. The use of aerial images and GPS for mobile robot waypoint navigation. *IEEE/ASME Trans Mechatron* 2008;13:692–99. DOI
34. Yang Y, Khalife J, Morales JJ, Kassas ZM. UAV waypoint opportunistic navigation in GNSS-denied environments. *IEEE Trans Aerosp Electron Syst* 2022;58:663–78. DOI
35. Teng F, Zhang H, Luo C, Shan Q. Delay tolerant containment control for second-order multi-agent systems based on communication topology design. *Neurocomputing* 2020;380:11–19. DOI
36. Lee DT, Drysdale RL III. Generalization of Voronoi diagrams in the plane. *SIAM J Comput* 1981;10:73–87. DOI
37. Takahashi O, Schilling RJ. Motion planning in a plane using generalized Voronoi diagrams. *IEEE Trans Robot* 1989;5:143–50. DOI
38. Lei T, Luo C, Ball JE, Rahimi S. A graph-based ant-like approach to optimal path planning. In: *IEEE Congress on Evolutionary Computation*. Glasgow, UK; 2020. pp. 1–6. DOI
39. Li X, Li X, Khyam MO, Luo C, Tan Y. Visual navigation method for indoor mobile robot based on extended BoW model. *CAAI Trans Intell Tech* 2017;2:142–47. Available from: <https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/trit.2017.0020> [Last accessed on 31 Aug 2022].
40. Yang Y, Deng Q, Shen F, Zhao J, Luo C. A shapelet learning method for time series classification. In: IEEE International Conference on Tools with Artificial Intelligence. San Jose, USA; 2016. pp. 423–30. DOI
41. Liu L, Luo C, Shen F. Multi-agent formation control with target tracking and navigation. In: IEEE International Conference on Information and Automation. Macao, China; 2017. pp. 98–103. DOI

42. Babinec A, Duchoň F, Dekan M, Pászto P, Kelemen M. VFH*TDT (VFH* with time dependent tree): A new laser rangefinder based obstacle avoidance method designed for environment with non-static obstacles. *Rob Auton Syst* 2014;62:1098–115. [DOI](#)
43. Zhang B, Jin W, Gao X, Chen W. A multi-goal global dynamic path planning method for indoor mobile robot. In: International Symposium on Robotics Intelligent Manufacturing Technology. Changzhou, China; 2021. pp. 97–103. [DOI](#)
44. Noormohammadi-Asl A, Taghirad HD. Multi-goal motion planning using traveling salesman problem in belief space. *Inf Sci* 2019;471:164–84. [DOI](#)
45. Zhuang H, Dong K, Qi Y, Wang N, Dong L. Multi-destination path planning method research of mobile robots based on goal of passing through the fewest obstacles. *Appl Sci* 2021;11:7378. [DOI](#)
46. Vonásek V, Pěnička R. Space-filling forest for multi-goal path planning. In: IEEE International Conference on Emerging Technologies and Factory Automation. Zaragoza, Spain; 2019. pp. 1587–90. [DOI](#)