Information Sciences 478 (2019) 318–330

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

An efficient approach to 3D path planning

Jihee Han

Department of Industrial Management Engineering, 145 Korea University, Anam-ro, Seoul, South Korea

ARTICLE INFO

Article history:
Received 8 April 2018
Revised 18 October 2018
Accepted 17 November 2018
Available online 19 November 2018

Keywords:
3D environment
Path planning
Point reduction
Surrounding point set

ABSTRACT

Since the environmental complexity of a 3D environment significantly increases as compared to a 2D environment, previous studies on 3D path planning have mostly focused on efficiency in generating a 3D path. In this context, the present study proposes a method, COSPS (Critical Obstacles and Surrounding Point Set), which can lower the complexity for efficient 3D path planning by determining a subset of obstacles in terms of their importance for planning a path, and by finding a subset of grid points that can fully surround those obstacles in 3D. Due to this reduction of obstacles and points, a path can be efficiently generated, as a graph is constructed with a smaller number of points, rather than with the total number of points in 3D. To observe the effect of the proposed method on 3D path planning as compared to well-known methods that consider all obstacles and points, the simulation experiment was conducted for multiple 3D maps with different sizes, shapes, and distribution of obstacles. The results demonstrated that the proposed method can generate a feasible path and significantly enhance the efficiency of 3D path planning without compromising the quality of the solution in terms of path length.

1. Introduction

Three-dimensional (3D) path planning is widely applicable in domains such as aerospace, oceanography, agriculture, and the military for many tasks, including but not limited to searching regions, space coverage, or reaching targets. With the proliferation of drones and UAVs (Unmanned Aerial Vehicle) that need to able to plan a feasible path in three dimensions, the applications of 3D path planning have become even more significant. In general, given the start and target locations, the goal of path planning is defined as planning a collision-free path from 3D obstacles while satisfying certain criteria, such as distance, smoothness, or safety. As mobile robot path planning is considered to be NP-hard, 3D path planning is also NP-hard with an additional axis for height [5,22,25]. In previous decades, several methods to overcome the complexity of this problem have been developed. The classical approaches include A*, Dijkstra, PRM (Probabilistic Roadmap Method), and Artificial Potential Field. These approaches were intensively studied in the early days of 3D path planning. More recently, path planning in three dimensions has been studied with heuristic approaches, such as soft computing, meta-heuristics, and hybridized heuristics with classical methods.

The importance of efficient mechanisms is higher for solving path planning problems in 3D space as compared to 2D space due to the added complexity [8,16]. This complexity stems from the increase in search space. If a path is represented by a set of points and segments between those points, the number of points to be considered to plan a path significantly increases in a 3D space. For instance, according to a grid map with the length of n for each axis and point by unit grid length, the total number of grid points increases to n^3 from n^2 in two dimensions. This also suggests that, if the points are

E-mail address: jrjrhan@gmail.com

https://doi.org/10.1016/j.ins.2018.11.045
0020-0255/

generated more densely, then the complexity in a 3D space would also increase. Other factors that increase the complexity of 3D path planning are uncertainty and dynamicity in environments, as they require data acquisition and processing over time [14,26]. Generally, as path planning in three dimensions is used in a larger space, sometimes across the world or even in space, unexpected and uncontrolled situations can occur. Consequently, paths might need to be modified in real time, requiring more efficient mechanisms of path planning.

Previous research using the classical methods of path planning in three dimensions can be summarized as follows. First, Yang and Sukkarieh [23] used the RRT (Rapidly-exploring Random Tree) technique to reduce the computational complexity of 3D path planning by developing a path pruning algorithm to remove unnecessary points. Furthermore, Hota and Ghose [8] focused on the geometry of 3D obstacles for planning a path, rather than on generating points in a map. As the A*, Dijkstra, Bellman Ford, and Floyd-Warshall algorithms have been widely used for path planning problems, one study compared the performance of those algorithms for a UAV application where all given points were considered in path planning [20]. In particular, the A* algorithm has been frequently used in previous research in terms of specific features of 3D environments, such as the orographic obstacles in urban environments considered with grid points [4]; in addition, unknown environments represented by grid points have also been studied where virtual terrain is introduced to reduce the search space from three to two dimensions [16].

As heuristic techniques have been studied recently in order to lower the computational cost, Aybars [2] used GA (Genetic Algorithm) for 3D path planning based on a set of pre-determined points on surfaces of a building. GA was also adopted in other studies to reduce the computational cost by combining it with either a Voronoi diagram to generate points in a 3D environment [15], or the 2-opt algorithm based on generating 8 points around the radar circle by 45° in threat environments [13]. Among the studies that considered the situation of moving obstacles and uncertainty of locations of obstacles, Rathbun et al. [18] used the evolutionary algorithm where paths are represented by intersection points between the curves of circles with a certain radius. In terms of multi-objective path planning, the parallel programming paradigm combining GA and PSO (Particle Swarm Optimization) was proposed for an efficient UAV path planning; this paradigm considered length, altitude, safety, fuel consumption, and smoothness in the objective function by taking all grid points into account [19]. Furthermore, in a study by Liu et al. [10], PSO was also used with all grid points, and the odor intensity method was introduced by putting different levels of odor intensity on candidate particles to quickly reach the goal. In addition, several other studies, such as [1,3,25], compared the methods, PSO, GA, ACO (Ant Colony Optimization), A*, RRT, memetic algorithm, and shuffled frog leaping algorithm, in terms of efficiency for the UAV path planning where points were generated by a grid, or by scattering in the space. In particular, Zeng et al. [25] and Miller et al. [12] projected three dimensions into two dimensions to resolve the complexity issue and to obtain efficient path planning.

Generally speaking, the classical methods and other approaches that consider all grid or scattered points are not appropriate methods for 3D path planning, because they are not able to successfully deal with high complexity. The soft computing techniques used to decrease the computational cost still have to face this problem, unless the environment is reduced for their iterative search mechanisms. Therefore, if all generated points are considered in path planning algorithms, then the complexity could not be easily resolved through those heuristic approaches, as the mechanisms are repeatedly conducted to explore the increased solution space. In cases of any random operations, the robustness of solutions is another issue. This is so because consistent outcomes would not be generated by each run. Furthermore, the approaches that obtain a path by reducing the problem to two dimensions would have a disadvantage, since doing so might lead to losing relevant path information. In addition, the approaches that allow for reaching neighboring points based on distance could also result in lower quality of solutions by increasing the path length, since the path segments are made locally at each step.

The present study proposes a method COSPS (Critical Obstacles and Surrounding Point Set) that decides a set of necessary obstacles and grid points for efficient 3D path planning. The background of the COSPS is that the high complexity of the 3D path planning problem is a result of searching unnecessary regions that do not play an important role in finding a path. Hence, to minimize unnecessary searching in 3D space, the proposed method is motivated by lowering the environmental information through focusing on partial areas that are determined to be more important than others. The overall steps of COSPS are to obtain the essential obstacles based on the importance for path planning and then to find grid points that fully surround those obstacles to take the least amount of grid points into account in generating a path. This approach resolves the issues of high complexity from the existing classical or heuristic methods in that these methods use all grid or scattered points in space for planning a 3D path, and resolves the issue of variation in solutions from soft computing techniques, since COSPS does not include any operations with randomness.

The contributions of the method proposed in the paper are summarized as follows:

i) A significant increase in the efficiency of path planning by reducing the number of obstacles and grid points in 3D environments;
ii) Obtaining a 3D path without decreasing the quality of the solution in terms of path length, even though only the limited map information is used for path planning;
iii) Achieving consistency in solutions due to the absence of randomness causing variations in the proposed method;
iv) High applicability of the method in various 3D environments by not requiring constraints on the shape of obstacles.

The remainder of this paper is organized as follows. Section 2 presents the definition and assumptions of 3D path planning. Section 3 outlines the proposed method to reduce the complexity of three dimensions. Simulation for the proposed method is conducted in Section 4, and conclusions are drawn in Section 5.

2. Problem definition and assumptions

This paper aims to develop a method that reduces the environmental complexity for efficient and effective 3D path planning. The approach finds a subset of obstacles and points in a 3D environment that are used for generating a feasible path from the start to the target points.

The environment is represented by three dimensions including map boundary, obstacles, points, and paths. As all obstacles are not subject to moving over time, and given the locations of start S, target T, and a set of obstacles O, a static environment and a known environment are considered. In addition, the environment is represented by a set of grid points N generated by three axes in terms of the straight line between S and T, and grid width w. As to the grid size, it is assumed to be sufficiently small to be placed between obstacles. The shapes of obstacles are not restricted, as these are defined by any coordinates in space, i.e. no approximation with grid points is required. The path is represented by feasible line segments between a set of points, $p \in N_f$, from S to T, and evaluated by length to obtain the shortest path.

Notations for the proposed method are summarized in Table 1

Table 1
Notations.

Notation	Description
N	Set of grid points
N_f	Set of grid feasible points, $N_f \subset N$
p	A grid point, $p \in N$
O	Set of obstacles
o_i	ith obstacle, $i \in \|O\|$
S	Starting point
T	Target point
w	Size of grid width

3. Methodology

In this section, the method for efficient 3D path planning is developed based on finding the points surrounding important obstacles. Research on CO and SPS previously proposed for 2D mobile robot path planning is summarized in Section 3.1, and the detailed development for 3D path planning is outlined in Section 3.2.

3.1. Previous research on the CO and SPS in 2D environment

Prior to the present work on 3D path planning, the CO and SPS concepts were originally developed for path planning for mobile robots in 2D environments [6]. To briefly review those concepts, the CO and SPS approach was inspired by the observation that the shortest paths would mostly be constructed around certain obstacles that directly collide with a straight line between the start and the target points, $\overline{ST}$. This is based on that the $\overline{ST}$ is the lower bound of the shortest path solution where no obstacle exists or lies in the way, and the obstacles that make the lower bound infeasible would be primarily responsible for constructing a feasible path [11]. In this regard, obstacles are not all considered to be the same level of importance but rather are classified into critical or ignorable obstacles based on the existence of intersection between obstacles and $\overline{ST}$. Thus the obstacles are defined as critical obstacles, CO, or non-critical obstacles, NCO [7]. After identifying CO, which is a subset of all obstacles, and given a map representation with grid points, the next step is to obtain a subset of grid points that completely surround the obstacles in CO; this subset is defined as SPS. Consequently, 2D path planning can be efficiently conducted using only a subset of obstacles and grid points.

3.2. The proposed approach to 3D path planning

Since the high complexity of 3D path planning results from considering all generated points in space, including those irrelevant to the final path, the proposed method focuses on removing the redundant points and obstacles to enhance the efficiency of path planning in a 3D environment. The method proposed here is developed from the previous study on CO and SPS described in Section 3.1. Given that, as compared to a 2D environment, a 3D environment has one additional axis, called height, and since obstacles are no longer polygons, but polyhedrons, the appropriate developments for the CO and SPS are proposed to apply those concepts to environments represented by three axes.

The proposed method begins by defining a set of critical obstacles, CO, and finding a set of surrounding points, SPS, which circumscribe the obstacles in CO. Then, the path is found in a graph generated with grid points on $\overline{ST}$ and in SPS, given a certain density. As the first step, CO and NCO, which is a set of non-critical obstacles, are identified from a set of obstacles, O, where $\mathrm{CO} \cup \mathrm{NCO} = O$ and $\mathrm{CO} \cap \mathrm{NCO} = \emptyset$, based on the collisions with $\overline{ST}$, since the obstacles colliding with $\overline{ST}$ play a more important role in path planning. Let I be a function that returns 1 based on the existence of an intersection between two inputs, and 0 otherwise. Then the CO and NCO can be defined as shown in Eqs. (1) and (2).

$$\mathrm{CO} = \{o_i | I(o_i, \overline{ST}) = 1, \ \forall o_i \in O\} \tag{1}$$

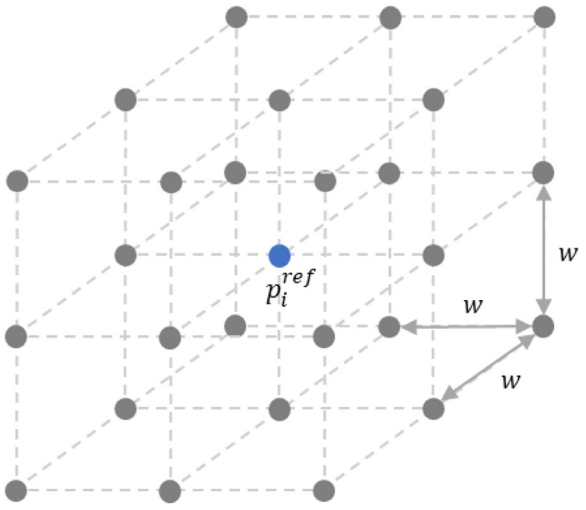


Fig. 1. A cube of a reference point.

$$\mathrm{NCO} = \{o_i | I(o_i, \overline{ST}) = 0, \ \forall o_i \in O\} \tag{2}$$

After identifying the CO that would be the obstacles selected for path planning, the SPS, which is a subset of N_f, is obtained to find a set of points surrounding those obstacles in CO. A set of points surrounding $o_i \in \mathrm{CO}$ is denoted as SPS_i; determining SPS_i is based on a certain reference point p_i^{ref}, its neighbor points that are the grid points within $\sqrt{3}w$ from p_i^{ref}, and a cube drawn by those neighbor points in three dimensions (see Fig. 1). The process is performed as a p_i^{ref} becomes an element of SPS_i if the cube of that p_i^{ref}, $cube_{p_i^{ref}}$, intersects with o_i. Then the feasible neighbors, $nb(p_i^{ref})$, defined as shown in Eq. (3), become elements of a tempset which is a waiting list containing reference points for the future SPS process. If there is no intersection between the $cube_{p_i^{ref}}$ and o_i, then p_i^{ref} is not determined as an element of SPS_i since its distance from o_i is not sufficiently close to be properly surrounding the obstacle. Consequently, the neighbor points of the p_i^{ref} are also ignored and not entered into the tempset. By repeating this process until the tempset becomes an empty set, the subset of grid points that surrounds o_i can be obtained.

$$nb(p_i^{ref}) = \{p | \left| p - p_i^{ref} \right| \le \sqrt{3}w, \ \ p \ne p_i^{ref} \ and \ \ p \in N_f\} \tag{3}$$

The initial reference point p_i^{ref} can be obtained as s_i, the starting point of SPS process for o_i. This allows the reference point to become a feasible grid point (see Proposition 1), and to prevent the tempset from being an empty set in the beginning, so that the SPS process could be initiated for o_i. In order to define s_i, where there are k number of intersecting points $\{ip_i^1, ip_i^2, \cdots, ip_i^k\}$ between $o_i \in O$ and $\overline{ST}$, the closest intersecting point from S is defined as c_i (see Eq. (4)). Given that a subset of grid points on $\overline{ST}$ is denoted as $N^{\overline{ST}} \subset N$, for a certain point q on $\overline{ST}$ which is not necessarily a grid point, let a subset of grid points from S to q on $\overline{ST}$ as $N_q^{\overline{ST}} \subset N^{\overline{ST}}$. Then s_i starts the SPS process for o_i (see Eq. (5)) in terms of c_i, which stands for the closest grid points from c_i among all grid points between S and c_i, $N_{c_i}^{\overline{ST}}$.

$$c_i = \underset{ip_i^k}{\mathrm{argmin}}\{\left| ip_i^k - S \right|, \ \forall k\} \tag{4}$$

$$s_i = \underset{p}{\mathrm{argmin}}\{|p - c_i|, \ \ p \in N_{c_i}^{\overline{ST}}\} \tag{5}$$

Proposition 1. *Assume that grid width w is sufficiently small for grid points to be placed between obstacles, then s_i is a feasible grid point ($s_i \in N_f$).*

Proof. For $i = 1$, since all points p on $\overline{sc_i}$ are feasible ($p \in N_{c_i}^{\overline{ST}} \subset N_f$), s_i is feasible ($s_i \in N_f$). For $1 < i \le |CO|$ where $|CO| > 1$, which implies that there is a previous obstacle, o_{i-1}, since s_i is defined as the closest grid point from c_i among all grid point on $\overline{sc_i}$, and there is a feasible grid point $\{p'\} \subset N_f$ between o_{i-1} and o_i by the assumption, s_i is feasible ($s_i \in \{p'\} \subset N_f$). □

An SPS_i surrounding $o_i \in \mathrm{CO}$ that starts from s_i is drawn with the blue circles in Fig. 2 where the grid points are marked by small grey dots. It shows that the SPS approach can find a set of grid feasible points in a 3D environment that fully surrounds an obstacle in a polyhedral shape (see Proposition 2).

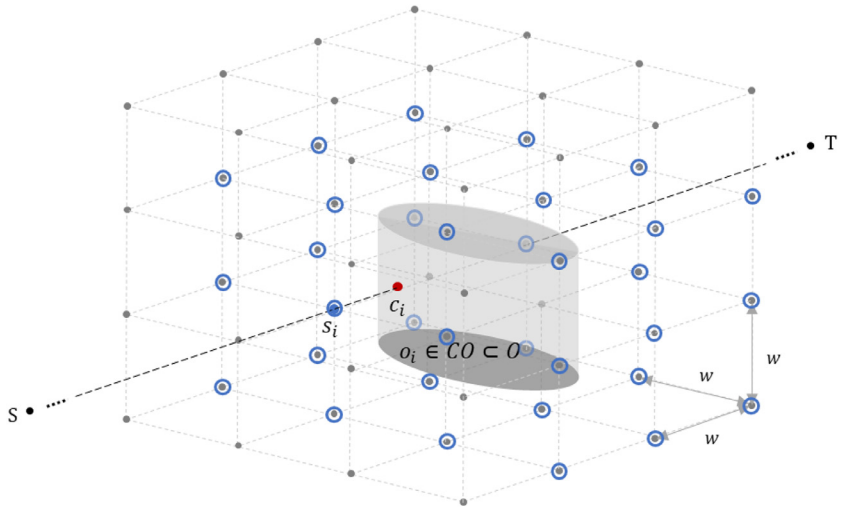


Fig. 2. The CO and SPS in a 3D environment.

Proposition 2. *Assume that grid width w is sufficiently small for grid points to be placed between obstacles, then* SPS_i *is a set of grid points fully surrounding* o_i.

Proof. Given a cube made from a certain grid feasible point $p \in N_f$ by w be denoted as $cube_p$ in three axes, let N_f be classified into $N_f^{I,i}$ and $N_f^{\sim I,i}$ based on the existence of an intersection between $cube_p$ and o_i. That is, $N_f = N_f^{I,i} \cup N_f^{\sim I,i}$ where $N_f^{I,i} = \{p | I(o_i, cube_p) = 1\}$ and $N_f^{\sim I,i} = \{p | I(o_i, cube_p) = o\}$.

Based on this division of N_f, we obtain $SPS_i \subset N_f^{I,i}$, since all points in SPS_i were moved from the tempset because their $cube_p$ intersects with o_i. Then, in order to show that SPS_i is a set of grid points that fully surround o_i, we show that $SPS_i = N_f^{I,i}$, because if the SPS process stops before finding all surrounding points by having an empty tempset, then it indicates that $SPS_i \neq N_f^{I,i}$.

Suppose for contradiction that there is a point $p' \in (N_f^{I,i} - SPS_i)$ indicating that $N_f^{I,i} - SPS_i \neq \emptyset$. Assume that there is a point $p'' \in (N_f^{I,i} \cap SPS_i)$ located within $\sqrt{3}w$ from p', $\|p'' - p'\| \leq \sqrt{3}w$. Then p' is an element of the neighbors of p'', $nb(p'')$. As p'' is an element of SPS_i indicating that its $nb(p'')$ goes into the tempset, p' becomes an element of the tempset. Then, p' moves from the tempset to SPS_i since it is an element of $N_f^{I,i}$, which contradicts $p' \in (N_f^{I,i} - SPS_i)$, finising the proof. □

After finding the sets of surrounding points for all obstacles in CO, a path from S to T can be planned. The path is generated based on a graph $G = (V, E)$ with V a set of vertices and E a set of edges connecting pairs of vertices, where V is obtained as the union of feasible grid points on $\overline{ST}$ and points in SPS through the proposed method (see Eq. (6)). As to the algorithms to construct a path, since the present paper proposes a method that reduces an environmental complexity in terms of obstacles and points, rather than suggests how to find a path in a network, any path planning related methods—such as the A*, Dijkstra, or Meta-heuristics algorithms—could be used to find a path.

$$V = \left(N^{\overline{ST}} \cap N_f\right) \cup SPS \tag{6}$$

In addition to enhancing the efficiency of 3D path planning, the proposed method has an advantage in terms of minimizing the path length. Specifically, in the proposed approach, 3D path planning considers a graph where each vertex connects to a certain number of vertices or the nearest neighbors, rather than to a full graph because the complexity of path planning significantly increases with an increase of the density of the graph. In this regard, where the density of a graph is defined as a limited number of connected points, a small number of points in the graph might be beneficial to reduce the path length. A simple example of this is shown in Fig. 3, assuming that the density value is set to 26, meaning that each grid point has edges with its 26 number of neighboring points. Comparing the two graphs, one (a) taking all feasible grid points N_f into account, and the other (b) considering only the subset of N_f by the proposed method (marked by blue circles), the example shows that the path length from point p to q is reduced as the values are calculated as $(\sqrt{3} + \sqrt{2} + 2)w \cong 5.14w$ in (a) and $\sqrt{21}w \cong 4.58w$ in (b). Even though many vertices are not allowed to be considered for path planning in graph (b), the practical assumption that constructs a graph within the limited density eventually lets the path be generated with a more global connection between the points.

The overall algorithm of the proposed method can be summarized as follows. Given the information about a 3D environment such as locations and shapes of obstacles, the start and the target points, and grid width, the map is represented

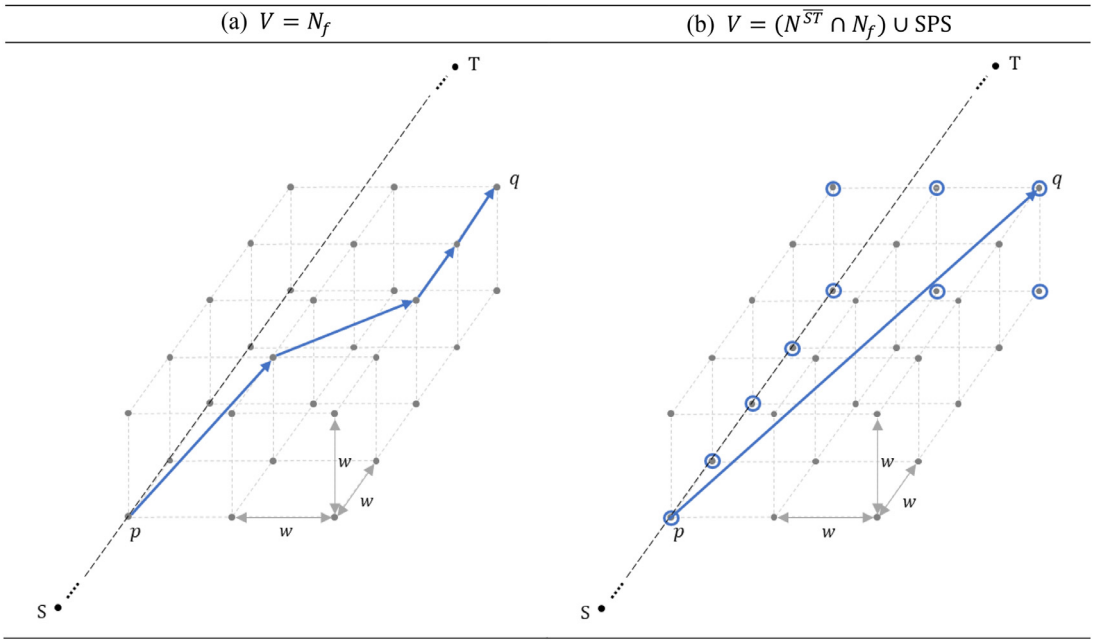


Fig. 3. Example of the path length reduction by the proposed approach.

Algorithm 1 COSPS for 3D path planning.

Input: O, S, T, and w
Generate a set of 3D grid points based on $\overline{ST}$ and grid width w
Identify a set of critical obstacles, CO based on intersection between O and $\overline{ST}$
for $o_i \in$ CO
  Find c_i from Eq. (4)
  Find s_i from Eq. (5)
  tempset $\leftarrow s_i$
  while tempset $\neq \emptyset$
    $p_i^{ref} \leftarrow$ tempset(1)
    if $I(cube_{p_i^{ref}}, o_i) == 1$
      $\text{SPS}_i = \text{SPS}_i \cup \{p_i^{ref}\}$
      tempset = tempset $\cup \{nb(p_i^{ref}) \setminus \{p | p \in \text{SPS}_i \text{ or } p \in \text{tempset}\}\}$
      tempset = tempset $\setminus$ {tempset(1)}
    else
      tempset = tempset $\setminus$ {tempset(1)}
    end
  end
  $\text{SPS} = \text{SPS} \cup \text{SPS}_i$
end
Generate a graph $G = (V, E)$ based on $V = (N^{\overline{ST}} \cap N_f) \cup \text{SPS}$
Find a path on the graph

by a set of grid points generated from $\overline{ST}$ and w. Based on the collision with $\overline{ST}$, a set of critical obstacles, CO, is identified, and the process to find the points surrounding those obstacles in CO is performed in the order of closeness from S. The reference point to start the SPS process is then determined, and the decision whether or not this reference point and its neighboring points go into SPS and the tempset, respectively, is made based on the existence of the intersection between the cube of the reference point and the obstacle. This procedure is iterated until the termination criterion is met, which is that the tempset should become an empty set. The process is repeated for all obstacles in CO. Then, the path from S to T is obtained where graph G is generated based on the grid points on $\overline{ST}$ and SPS (Algorithm 1).

4. Simulation

In this section, the simulation is conducted to test the proposed method in terms of its capacity to reduce the computational complexity and obtaining a feasible path in three dimensions. With the variation of grid width values, the results are discussed with respect to the effectiveness of the proposed method in dealing with an increase of the number of points.

4.1. Simulation design

To test the performance of the proposed method, six maps with different features were simulated with the boundaries of maps defined as those of a cube sized 2000 (cm) for each of the axes (x, y, z) as (see Table 2). Maps were constructed to

Table 2
Map information.

Map no.	$\|O\|$	Map size (cm)	S (x,y,z)	T (x,y,z)
1	18	$2000 \times 2000 \times 2000$	(206.49, 1861.52, 97.06)	(1729.25, 224.06, 1971.50)
2	14	$2000 \times 2000 \times 2000$	(835.07, 206.06, 692.96)	(1169.67, 1858.28, 1081.54)
3	3	$2000 \times 2000 \times 2000$	(181.47, 690.58, 712.70)	(1891.34, 1763.24, 1209.75)
4	5	$2000 \times 2000 \times 2000$	(203.53, 297.18, 449.16)	(1651.35, 1005.01, 1110.63)
5	24	$2000 \times 2000 \times 2000$	(665.82, 1903.78, 103.20)	(1863.32, 302.66, 1837.75)
6	12	$2000 \times 2000 \times 2000$	(232.38, 251.20, 1029.58)	(1003.21, 1342.88, 1038.47)

apply the proposed method in various environments, such as aerospace, city, a terrain with mountains, and a maze, so that the applicability of the method could be established through the simulation. The shapes of the obstacles were different across the maps, where the six maps contained random shapes of polyhedrons floating in the air, buildings of different heights on the ground, cylinders with holes in the middle, several peaks, and perpendicular planes, respectively. The range of the number of obstacles, $|O|$, was from 3 to 24, and the locations of these obstacles were randomly distributed in the maps, including both dense and sparse cases. The coordinates of the start and the target were also randomly generated, maximally covering the regions of the maps.

After determining a set of obstacles and nodes considered for path planning, a 3D graph was generated based on the density value, d, which represented the number of neighbor points connected to each point. The value of d is set to be one percent of all grid feasible points, $|N_f|$. In addition, the Dijkstra algorithm [17] was applied as a method for planning the shortest path in a given graph.

In order to compare the results of the proposed method, the well-known methods PRM and Wavefront algorithm were also simulated, as these methods have been widely used for 3D path planning. In addition, these well-known methods are suitable to be chosen as no-reduction approaches, because they do not adopt removal of obstacles and points in three dimensions. PRM randomly scatters a set of points in configuration space and uses a pathfinding procedure to plan a path with those points. Since PRM does not adopt the grid map representation, a path that uses the free coordinates in configuration space can be generated [9]. On the other hand, Wavefront algorithm, also called numerical potential field, assigns an increasing value by 1 from target on each neighbor grid point, and finds a path by connecting the points from start to the neighboring point that has the steepest decent. This approach quickly generates a path, as it only connects to the neighboring grid points [21,24].

Simulations were conducted with the same number of points for all three approaches (PRM, Wavefront algorithm, and COSPS) to see how these methods perform in generating a 3D path. The Dijkstra algorithm was used to find the shortest path in PRM network, and neighbor points were defined as the points within the grid width in Wavefront algorithm. For the performance test, the width w was decreased from 200 to 100 by 25, aiming to figure out how effectively the proposed method could handle the complexity with an increase of the number of points in 3D maps. All simulations were implemented 10 times to obtain the average and standard deviation in solutions.

4.2. Simulation results

According to the difference in the considered obstacles and points, the simulation results of the proposed method, PRM, and Wavefront algorithm are summarized by maps and w values in Table 3 and Fig. 4. With a decrease of w from 200 to 100, the graphs are more densely represented in places where the number of points increased from 770 at $w = 200$ to 8004 at $w = 100$. The results including paths, obstacles, and considered points are visualized in 3D maps, where the first, second, and third rows represent the PRM, Wavefront algorithm and the proposed approach, respectively (see Fig. 4). All obstacles are drawn as polyhedrons in the grey color, and the considered points for path planning are depicted by the grey dots. The final paths are drawn with the blue lines.

As shown in Fig. 4, as compared to the no-reduction approaches (PRM and Wavefront algorithm), the COSPS decreased the number of obstacles and considered points for all maps and w values. In terms of obstacles, only 4, 3, 1, 2, 4, and 1 obstacles were considered by the proposed method, while the no-reduction approaches considered all 18, 14, 3, 5, 24, and 12 obstacles. The smallest number of points was also obtained by the COSPS, showing that on average 2521.93 fewer points were considered for path planning when this method was used. This result indicates that taking all generated obstacles and points for path planning would not be an appropriate way to deal with a 3D environment because there are many points in 3D maps that are irrelevant for constructing a path. According to the different features of the maps, the reduction of the number of points was the minimum of 537 points at $w = 200$ in map 5 and the maximum of 7515 points at $w = 100$ in

Table 3
Summary of simulation results for the three approaches.

				PRM		Wavefront		COSPS			
Map No.	w	$\|O\|$	No. of points	Path length	CPU time	Path length	CPU time	Reduced $\|O\|$	Reduced No. of points	Path length	CPU time
1	200	18	982	$3428.28_{73.49}$	$21.31_{1.58}$	$3317.81_{0.00}$	$14.65_{0.08}$	4	155	$3229.90_{0.00}$	$4.50_{0.21}$
	175		1470	$3266.53_{53.79}$	$45.87_{3.41}$	$3617.81_{0.00}$	$22.42_{0.47}$		192	$3136.50_{0.00}$	$7.86_{0.32}$
	150		2330	$3148.82_{40.63}$	$106.42_{1.65}$	$3517.81_{0.00}$	$35.87_{0.36}$		226	$3070.69_{0.00}$	$12.94_{0.19}$
	125		4043	$3074.96_{25.08}$	$324.44_{4.53}$	$3417.81_{0.00}$	$63.89_{0.24}$		284	$3003.24_{0.00}$	$26.95_{0.64}$
	100		7884	$3009.40_{16.18}$	$1353.80_{4.58}$	$3317.81_{0.00}$	$128.47_{0.58}$		369	$2965.45_{0.00}$	$61.64_{0.76}$
2	200	14	918	$2529.20_{190.68}$	$13.22_{0.12}$	$2529.97_{0.00}$	$10.85_{0.24}$	3	289	$2321.36_{0.00}$	$6.03_{0.23}$
	175		1352	$2434.37_{118.21}$	$30.52_{0.95}$	$2429.97_{0.00}$	$16.12_{0.10}$		365	$2273.70_{0.00}$	$10.04_{0.19}$
	150		2145	$2198.06_{74.70}$	$76.27_{1.07}$	$2629.97_{0.00}$	$26.26_{0.05}$		504	$2128.84_{0.00}$	$19.42_{0.28}$
	125		3781	$2116.08_{37.01}$	$232.50_{2.85}$	$2479.97_{0.00}$	$47.60_{0.14}$		694	$2112.55_{0.00}$	$43.77_{0.71}$
	100		7268	$2068.11_{28.29}$	$937.05_{2.75}$	$2529.97_{0.00}$	$94.60_{0.10}$		1063	$2037.97_{0.00}$	$110.65_{1.20}$
3	200	3	939	$2887.53_{183.71}$	$5.79_{0.13}$	$2878.77_{0.00}$	$3.01_{0.07}$	1	249	$2463.48_{0.00}$	$3.39_{0.11}$
	175		1394	$2516.66_{107.05}$	$13.48_{0.24}$	$2778.77_{0.00}$	$4.49_{0.12}$		321	$2368.72_{0.00}$	$4.96_{0.12}$
	150		2222	$2350.78_{83.52}$	$34.61_{1.45}$	$2678.77_{0.00}$	$7.26_{0.16}$		410	$2276.24_{0.00}$	$8.58_{0.17}$
	125		3836	$2282.22_{31.86}$	$108.82_{2.11}$	$2578.77_{0.00}$	$13.09_{0.31}$		544	$2212.53_{0.00}$	$16.95_{0.33}$
	100		7443	$2224.45_{10.24}$	$511.42_{6.18}$	$2678.77_{0.00}$	$26.35_{0.10}$		830	$2247.10_{0.00}$	$43.28_{0.72}$
4	200	5	770	$2386.97_{225.23}$	$6.41_{0.17}$	$2542.05_{0.00}$	$4.82_{0.22}$	2	161	$2190.58_{0.00}$	$3.73_{0.02}$
	175		1156	$2145.61_{113.61}$	$14.31_{0.58}$	$2442.05_{0.00}$	$7.25_{0.20}$		222	$1998.27_{0.00}$	$5.82_{0.17}$
	150		1853	$1974.44_{71.30}$	$36.42_{0.78}$	$2342.05_{0.00}$	$12.10_{0.28}$		305	$1937.14_{0.00}$	$10.58_{0.19}$
	125		3192	$1926.47_{40.30}$	$110.91_{0.76}$	$2242.05_{0.00}$	$21.04_{0.20}$		413	$1860.09_{0.00}$	$17.87_{0.08}$
	100		6232	$1849.43_{25.35}$	$478.57_{6.08}$	$2342.05_{0.00}$	$43.32_{0.38}$		623	$1837.83_{0.00}$	$40.59_{0.37}$
5	200	24	981	$3716.50_{193.61}$	$24.49_{0.04}$	$4246.94_{0.00}$	$19.60_{2.11}$	4	444	$3297.67_{0.00}$	$13.07_{0.40}$
	175		1448	$3291.88_{123.64}$	$52.94_{1.94}$	$4046.94_{0.00}$	$28.16_{0.18}$		564	$3059.03_{0.00}$	$21.56_{0.17}$
	150		2316	$3133.52_{87.87}$	$133.62_{1.71}$	$3846.94_{0.00}$	$47.07_{0.71}$		771	$3041.49_{0.00}$	$43.68_{0.71}$
	125		4007	$2998.25_{65.67}$	$397.82_{2.01}$	$3646.94_{0.00}$	$84.53_{0.25}$		1123	$3047.98_{0.00}$	$102.47_{0.15}$
	100		7824	$2936.79_{54.44}$	$1574.21_{3.31}$	$3646.94_{0.00}$	$173.64_{1.03}$		1738	$3024.56_{0.00}$	$287.14_{0.75}$
6	200	12	1001	$7488.97_{187.31}$	$9.14_{0.41}$	$8936.42_{0.00}$	$7.33_{0.19}$	1	590	$7189.06_{0.00}$	$8.54_{0.21}$
	175		1433	$6758.03_{138.66}$	$18.02_{0.91}$	$7986.42_{0.00}$	$11.37_{0.26}$		724	$6215.96_{0.00}$	$14.44_{0.28}$
	150		2320	$6322.74_{38.13}$	$48.15_{1.77}$	$7936.42_{0.00}$	$18.99_{0.48}$		979	$6193.10_{0.00}$	$27.08_{0.37}$
	125		4076	$6126.38_{31.91}$	$155.82_{3.20}$	$8086.42_{0.00}$	$33.91_{1.31}$		1491	$6099.05_{0.00}$	$64.29_{0.76}$
	100		8004	$5887.45_{15.37}$	$627.56_{4.95}$	$7936.42_{0.00}$	$70.40_{2.19}$		2319	$5984.66_{0.00}$	$176.17_{0.85}$

map 1. This result is due to map 5 consisting of obstacles that are thin but have wide surfaces, thus leading to a larger number of points surrounding the obstacles in CO as compared to other maps. On the contrary, as map 1 has obstacles represented with relatively small surfaces, relatively fewer points are required to fully surround the obstacles in CO.

With respect to the path, the proposed approach successfully found feasible paths for all maps and w values, even though the graphs were made based on only a subset of obstacles and grid points. Along with achieving feasible solutions, the results showed that COSPS generated the shortest paths on average for all maps. The path length was reduced in COSPS, because edges between distant points could be connected in the graph and, consequently, the advantage of having a small number of points in reducing the path length was confirmed. However, when the map consisted of a large number of points with small sized grid widths, COSPS planned longer paths, as compared to the method that considers all points in the map. As shown in maps 3, 5, and 6 of Table 3 and Fig. 4, the shortest paths were achieved by COSPS up to $w = 125$ from 200 for maps 3 and 6, and $w = 150$ from 200 for map 5, while PRM generated shorter paths at smaller w. This suggests that COSPS would not have path segments that form the shortest path, since edges between points were relatively short when points are densely in the graph constructed with a limited number of edges. The longest paths were obtained by the Wavefront algorithm, since its mechanism is based on the perpendicular connection between neighbor grid points within the grid width. PRM had the highest standard deviation of path length because, unlike Wavefront algorithm and COSPS that use grid map, it plans a path based on randomly generated points; therefore, different paths are generated by each simulation run.

Concerning the CPU runtime, the COSPS and Wavefront algorithm achieved a much faster computing time as compared to PRM. The average runtime values were 40.60, 36.62, and 250.13 (s) for COSPS, Wavefront, and PRM, respectively. PRM had with the highest computational cost, since it constructed a complex graph by taking all points into account, and all the edges were checked for feasibility with obstacles. In this regard, the computational cost is more significant when dealing with many obstacles. By contrast, COSPS had a smaller number of points for path planning than the other two approaches, and thus a shorter runtime was required. The results also confirmed that the number of points plays an important role in computational cost, since maps 2, 5 and 6 determined a relatively large number of surrounding points and required a longer runtime than other maps. For Wavefront algorithm, even though it is considered as an inexpensive method that allows to simply connect the neighbor grid points based on the values propagated from the target, it required a long runtime to test feasibility for a large number of edges, since it considered all grid points for path planning. The number of obstacles also played a critical role in runtime (see maps 1, 2, and 5), which had many more of obstacles than the other maps, thereby increasing computational costs.

Fig. 5 shows the performance results of the three approaches as a function of decreasing w (an increase of number of points) on average of all six maps. In the left plot, as the maps are more densely represented with points, PRM and

Wavefront algorithm show a large increase of the number of points considered from $w = 200$ to 100, which is 6510.67 on average. On the other hand, the proposed approach shows a minor increment of 842.33 points by determining an essential set of obstacles and grid points. According to this reduction value on the number of points, it can be concluded that as a 3D environment becomes more complicated with an increase of the number of points, the effectiveness in dealing with the large search space is enhanced by the COSPS method, because the number of considered points increases more slowly than the total number of points.

In the center plot, which is the average path length at each w value, the COSPS shows a convergence of quality of solution as the number of points increases due to a reduction of the path length at a decreasing rate. Wavefront also has a decrease of path length with a decrease of w values, but a similar level of convergence to the COSPS is not achieved due to its perpendicular connection mechanism between neighboring grid points. For PRM, by having steeper slopes than other approaches, the path length converged to a value similar to the value for COSPS. The differences in path length between PRM and COSPS became smaller from 290.90 at $w = 200$ to -20.32 (cm) at $w = 100$ on average. This result of no significant difference in path length indicates that not considering full information about obstacles and points does not lower the quality of the solution.

When planning a path in low complexity, all three approaches have similar CPU runtime values (see the right plot). However, as the 3D environments become more complex with an increase of the number of points, PRM has the steepest increase from 13.39 to 913.77 (s) as compared to the other two approaches, where 6.54 to 119.91 (s), and 10.04 to 89.46 (s) for COSPS and Wavefront algorithm, respectively. Therefore, PRM is the weakest approach for solving path planning problems in complex environments. For COSPS and Wavefront algorithm, a similar trend of a slight increase in the runtime with a decrease of w is observed. This result implies that, even when compared to the approach that uses a simple connection method, COSPS achieves a similar level of computational cost by preventing the cost being exponentially large for searching 3D space.

The simulation shows that the proposed method can generate a feasible path from S to T in various 3D environments, even though the entire set of points is not considered. Compared to two different well-known, no-reduction methods, COSPS obtains the solutions much more efficiently than PRM and plans shorter paths than Wavefront algorithm for all maps and w

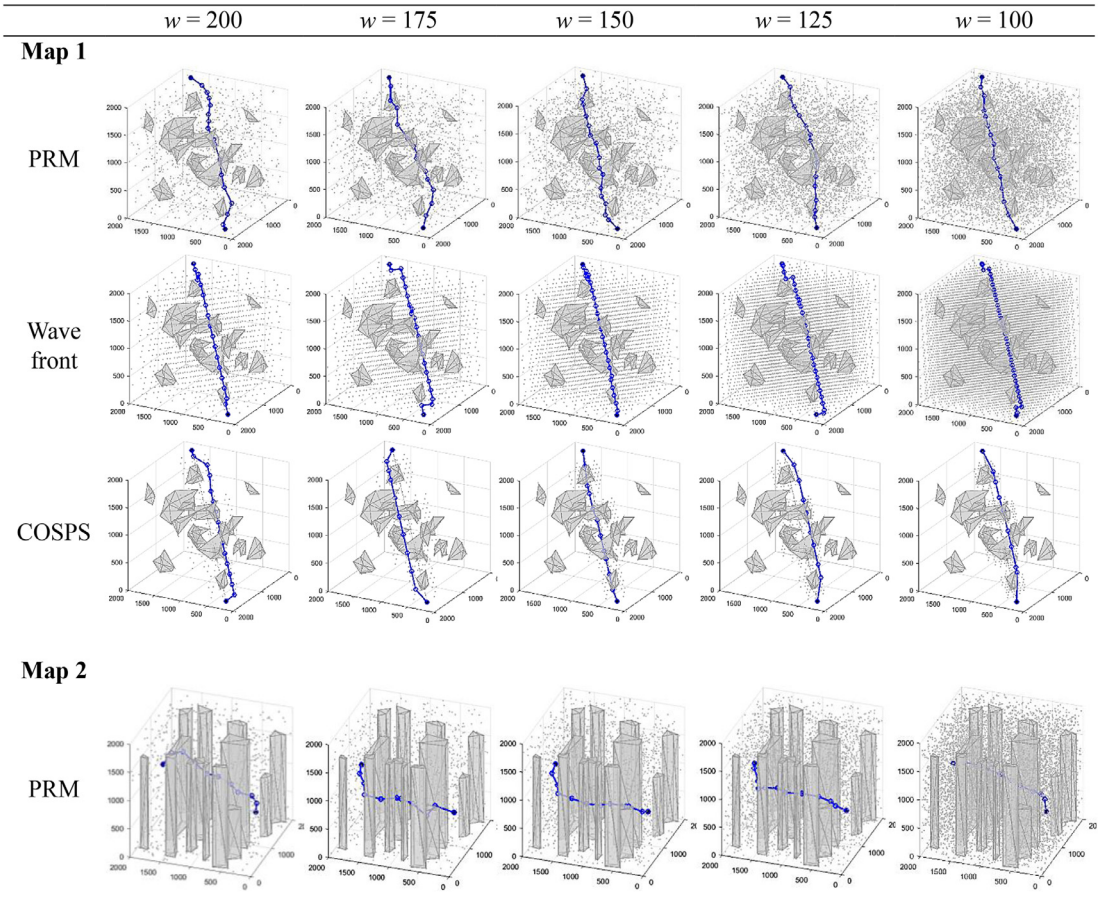
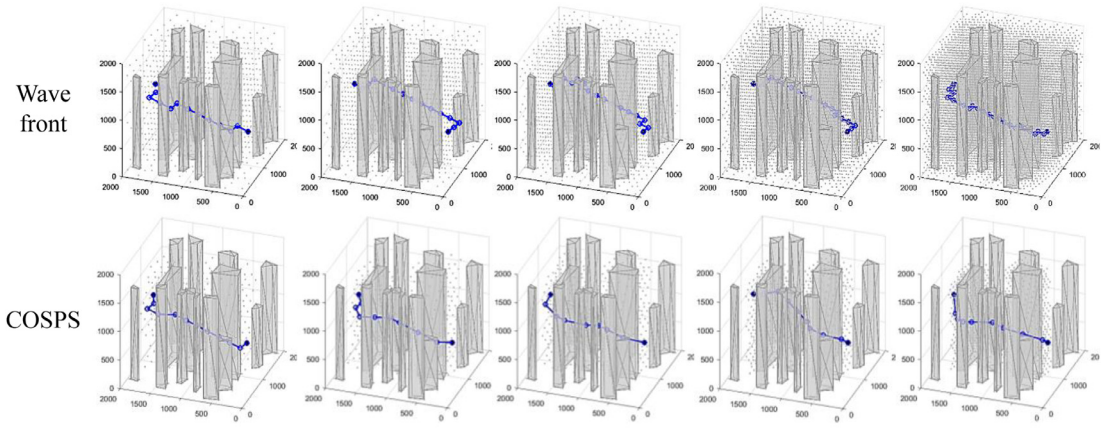
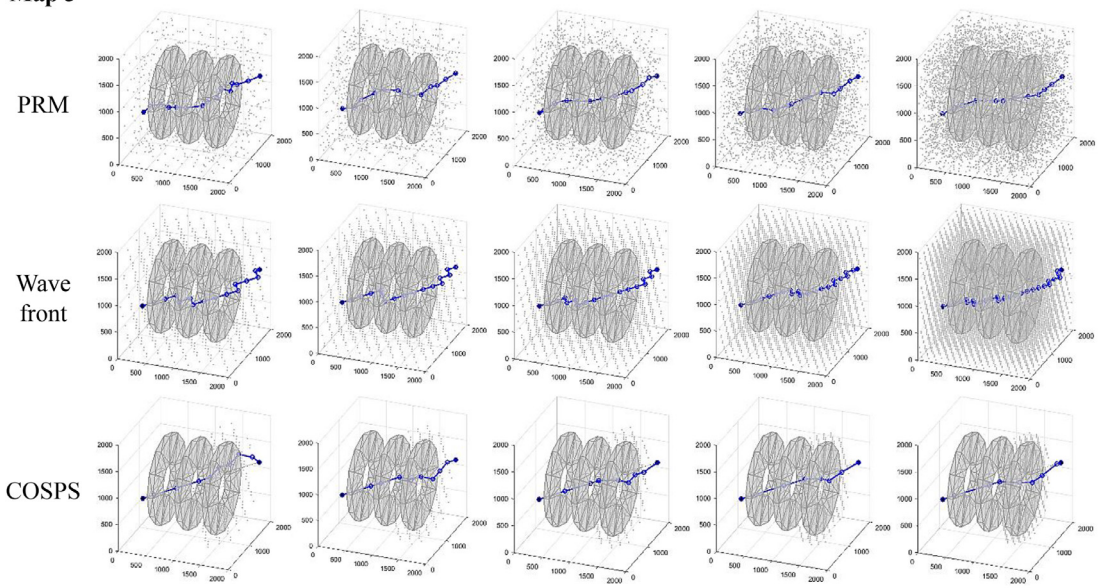


Fig. 4. Result plots in terms of obstacles, points, and the path.



Map 3



Map 4

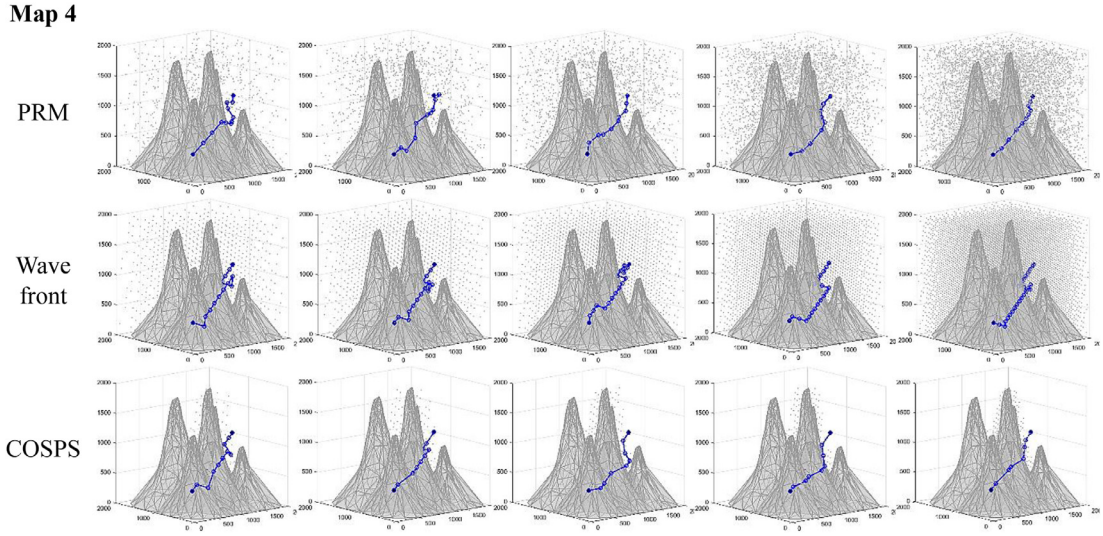


Fig. 4. Continued

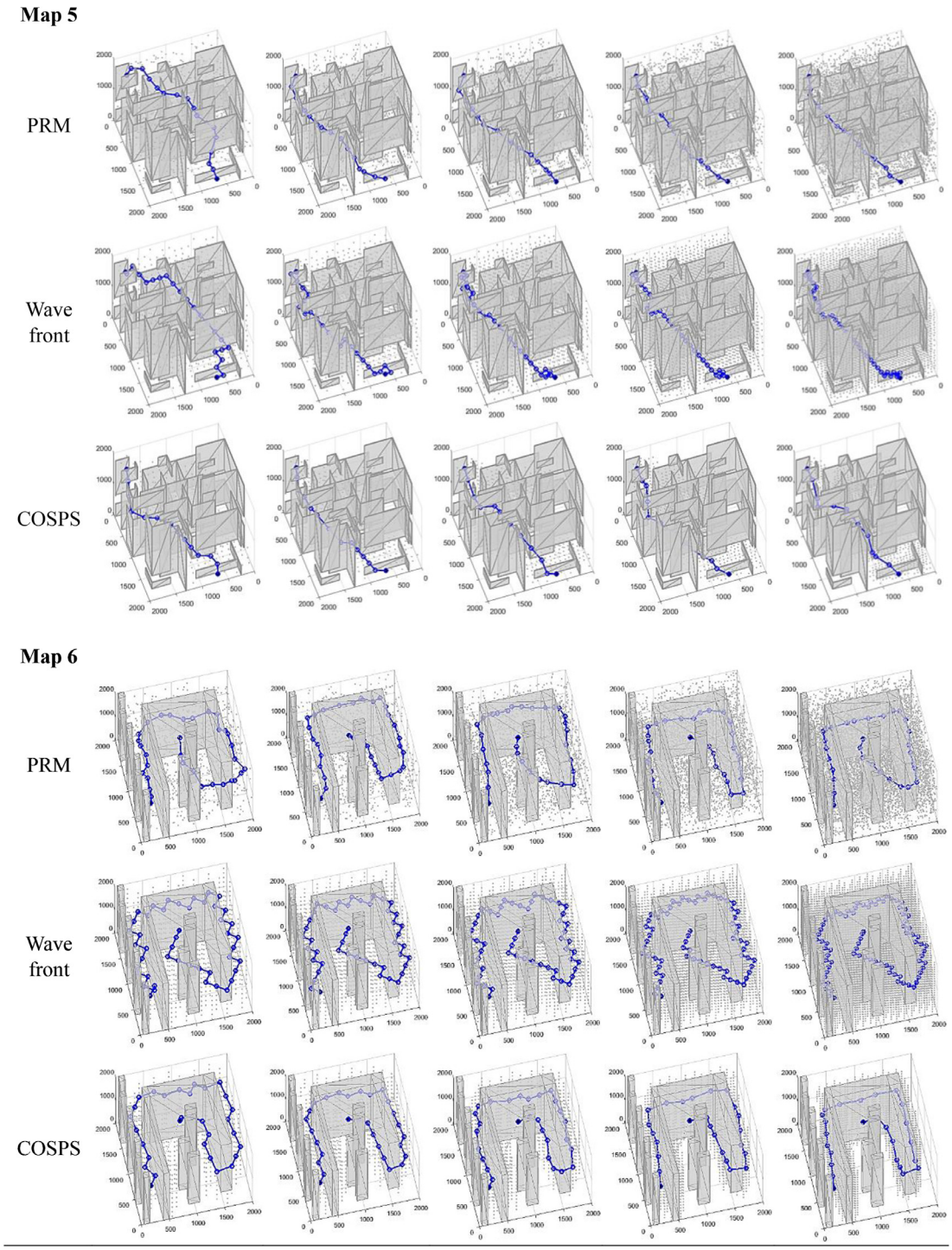


Fig. 4. Continued

values. Moreover, the simulation results demonstrate that the efficiency of path planning by the proposed method becomes even more significant as the point density increases in 3D map. Hence, the proposed approach can be successfully applied to 3D path planning, efficiently handling environmental complexity, and accomplishing a similar level of solution quality as the methods that use all 3D map information.

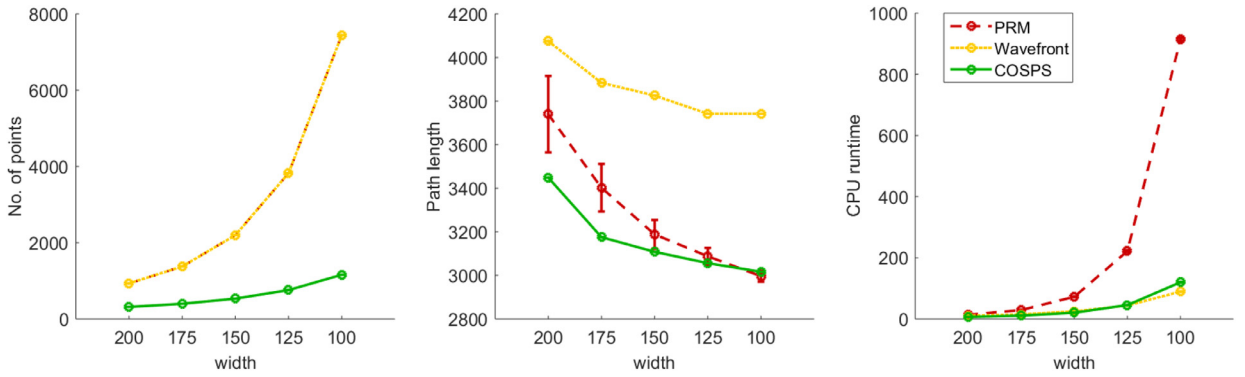


Fig. 5. Performance results with a decrease of w.

5. Conclusion

The present study developed an efficient 3D path planning method that resolves the issue of high complexity by adopting an essential subset of obstacles and grid points in three dimensions. As shown by the results of the simulation experiment, as compared to the paths obtained from the methods that do not reduce the number of obstacles and points considered, the proposed method successfully found a feasible path and achieved a significant efficiency in 3D path planning without compromising the quality of the solution in terms of length.

The proposed method can be used for any applications requiring 3D path planning. In particular, when a path is frequently modified in real time, since the efficiency becomes an important factor, the approach that does not consider unnecessary regions of the map would contribute to reducing the computational cost and accomplishing the tasks without having efficiency-related failures.

In future research, the proposed method could be further developed for dynamic and unknown environments where 3D obstacles are moving, or where the map is partially recognized based on sensor data, rather than when full map information is available. In 3D environments, as compared 2D, there can be more unexpected situations, such as threats and danger zones, which require real-time path planning. Further development that would consider those unknown and dynamic environments would be an important extension of the method presented herein.

References

[1] M.P. Aghababa, 3D path planning for underwater vehicles using five evolutionary optimization algorithms avoiding static and energetic obstacles, Appl. Ocean Res. 38 (2012) 48–62.
[2] U. Aybars, Path planning on a cuboid using genetic algorithms, Inf. Sci. 178 (2008) 3275–3287.
[3] E. Besada-Portas, L. De La Torre, A. Moreno, J.L. Risco-MartíN, On the performance comparison of multi-objective evolutionary UAV path planners, Inf. Sci. 238 (2013) 111–125.
[4] L. De Filippis, G. Guglieri, F. Quagliotti, Path planning strategies for UAVS in 3D environments, J. Intell. Rob. Syst. 65 (2012) 247–264.
[5] H. Duan, P. Qiao, Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning, Int. J. Intell. Comput. Cybern. 7 (2014) 24–37.
[6] J. Han, Y. Seo, Mobile robot path planning with surrounding point set and path improvement, Appl. Soft Comput. 57 (2017) 35–47.
[7] J. Han, Y. Seo, Path regeneration decisions in a dynamic environment, Inf. Sci. 450 (2018) 39–52.
[8] S. Hota, D. Ghose, Optimal path planning for an aerial vehicle in 3D space, in: 2010 49th IEEE Conference on Decision and Control (CDC), IEEE, 2010, pp. 4902–4907.
[9] S. Hrabar, 3D path planning and stereo-based obstacle avoidance for rotorcraft UAVs, in: 2008. IROS 2008. IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2008, pp. 807–814.
[10] Y. Liu, X. Zhang, X. Guan, D. Delahaye, Potential odor intensity grid based uav path planning algorithm with particle swarm optimization approach, Math. Probl. Eng. 2016 (2016).
[11] V.J. Lumelsky, A.A. Stepanov, Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape, Algorithmica 2 (1987) 403–430.
[12] B. Miller, K. Stepanyan, A. Miller, M. Andreev, 3D path planning in a threat environment, in: 2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC), IEEE, 2011, pp. 6864–6869.
[13] N. Özalp, O.K. Sahingoz, Optimal UAV path planning in a 3D threat environment by using parallel evolutionary algorithms, in: 2013 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, 2013, pp. 308–317.
[14] A.A. Paranjape, K.C. Meier, X. Shi, S. Chung, S. Hutchinson, Motion primitives and 3D path planning for fast flight through a forest, Int. J. Rob. Res. 34 (2015) 357–377.
[15] Y.V. Pehlivanoglu, A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV, Aerosp. Sci. Technol. 16 (2012) 47–55.
[16] Z. Qi, Z. Shao, Y.S. Ping, L.M. Hiot, Y.K. Leong, An improved heuristic algorithm for UAV path planning in 3D environment, in: 2010 2nd International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), IEEE, 2010, pp. 258–261.
[17] R.L. Rardin, Optimization in Operations Research, Prentice Hall, 2016.
[18] D. Rathbun, S. Kragelund, A. Pongpunwattana, B. Capozzi, An evolution based path planning algorithm for autonomous motion of a UAV through uncertain environments, in: Proceedings of the 21st Digital Avionics Systems Conference, 2002, IEEE, 2002 8D2-8D2.
[19] V. Roberge, M. Tarbouchi, G. Labonté, Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning, IEEE Trans. Ind. Inf. 9 (2013) 132–141.
[20] B.M. Sathyaraj, L.C. Jain, A. Finn, S. Drake, Multiple UAVs path planning algorithms: a comparative study, Fuzzy Optim. Decis. Making 7 (2008) 257.

[21] F. Schøler, 3d Path Planning for Autonomous Aerial Vehicles in Constrained Spaces, in Section of Automation & Control, Department of Electronic Systems, Aalborg University, 2012.
[22] K. Sugihara, J. Smith, Genetic algorithms for adaptive motion planning of an autonomous mobile robot, in: Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation, 1997 (CIRA'97), IEEE, 1997, pp. 138–143.
[23] K. Yang, S. Sukkarieh, 3D smooth path planning for a UAV in cluttered natural environments, in: IEEE/RSJ International Conference on Intelligent Robots and Systems 2008, IROS 2008, IEEE, 2008, pp. 794–800.
[24] L. Yang, J. Qi, D. Song, J. Xiao, J. Han, Y. Xia, Survey of robot 3D path planning algorithms, J. Control Sci. Eng. 2016 (2016) 5.
[25] Z. Zeng, K. Sammut, L. Lian, F. He, A. Lammas, Y. Tang, A comparison of optimization techniques for AUV path planning in environments with ocean currents, Rob. Autom. Syst. 82 (2016) 61–72.
[26] Y. Zhao, Y. Liu, Z. Li, Z. Duan, Distributed average tracking for multiple signals generated by linear dynamical systems: an edge-based framework, Automatica 75 (2017) 158–166.