Automation in Construction 95 (2018) 275–283

Contents lists available at ScienceDirect

Automation in Construction

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

Universal path planning for an indoor drone

Fangyu Li[a,b,*], Sisi Zlatanova[c], Martijn Koopman[d], Xueying Bai[e], Abdoulaye Diakité[c,**]

[a] College of Geosciences, China University of Petroleum, 18 Fuxue Road, Changping, Beijing 102249, China
[b] State Key Laboratory of Petroleum Resources and Prospecting, China University of Petroleum, Beijing 102249, China
[c] The University of New South Wales, UNSW, Sydney, NSW 2052, Australia
[d] CGI Nederland BV, Rotterdam, the Netherlands
[e] University of Southern California, Los Angeles, CA, USA

ARTICLE INFO

Keywords:
Path planning
Indoor
Drone
Distance transform
A* algorithm

ABSTRACT

During the construction and maintenance of building, universal path planning for an indoor drone navigation is needed. There are many two-dimensional (2D) path planning methods, but they are not appropriate for a three-dimensional (3D) indoor environment with many obstacles in it. In this study, we present a novel approach to plan universal paths for drones in a known indoor environment using a voxel model. This approach can make the drone fly at some distance from the obstacles by computing a 3D buffer around the obstacles, using our algorithm 3D propagating approximate Euclidean distance transformation (3D PAEDT). Two types of paths are presented using A* and distance transformation algorithms: *safe shortest path* (SSP) and *safe least cost path* (SLCP). Both paths ensure that the drone maintains a minimal distance from the obstacles. SLCP ensures that the drone flies at a fixed height. Several experiments are conducted to test the approach in a two-story building.

1. Introduction

A drone, also called an unmanned aerial vehicle, is an aircraft without a human pilot aboard. Drones have become smaller, lighter, and cheaper with the development of new technologies in recent years [1,2]. This makes many drone-based indoor navigation applications possible to automate construction process, such as information gathering, taking photos, surveying construction sites, monitoring construction progress [3,4], inspecting built infrastructure [5], and emergency help and rescue. Autonomous indoor navigation has become a foreseeable trend for drones because it saves the operator's time during complicated operations [6–10]. Autonomous indoor navigation functions include localizing, detecting obstacles, path planning, direction correcting, and tracking the trajectories with reasonable accuracy. Path planning is one of the main and fundamental aspects of autonomous navigation. Therefore, it is important to further extend indoor path planning algorithms and methods for autonomous drones.

Indoor path finding for a drone can be subdivided into real-time path finding in an unknown environment [6,9,10] and path finding in a priori-known environment [7,8]. With the technology developments for indoor data acquisition and three-dimensional (3D) indoor modeling, indoor data can be obtained quickly and easily and 3D indoor models with obstacles are increasingly made available [11,12]. Such models are a good basis to investigate approaches for universal path planning. Universal path planning, which can be seen as a key component of the real-time navigation, provides a collision-free continuous path, connecting the start and the goal (target) in a known environment. The current methods for people and mobile robots are commonly based on two-dimensional (2D) floor plans and indoor surface models [13,14]. However, drones fly in the air and can move above and below obstacle (e.g. furniture). There are many static obstacles on the floor, above the floor and near the ceiling, which have to be avoided. Two-dimensional approaches cannot overcome challenges such as “moving under or above obstacles”, “flying at a certain height”, or “avoiding overhanging parts”. Thus, it is necessary to consider 3D paths for drones, considering specific indoor requirements. Furthermore, it is critical that the computed path is safe, i.e. a certain clearance from all surrounding obstacles has to be ensured. These challenges are poorly addressed in the literature. Most previous studies have focused on optimal path planning for mobile robots. Unfortunately, optimal paths tend to “graze” the obstacles to minimize path lengths. Only few studies consider obstacles for indoor path planning [12].

The contribution of this study is an algorithm that achieves safe path planning by calculating a 3D buffer around the obstacles, based on propagation distance transformation. This approach keeps the drone from flying too close to the obstacles and sets the minimal distance from

* Correspondence to: F. Li, China University of Petroleum, 18 Fuxue Road, Changping, Beijing 102249, China.
** Corresponding author.
E-mail addresses: lifangyu@cup.edu.cn (F. Li), s.zlatanova@unsw.edu.au (S. Zlatanova), a.diakite@unsw.edu.au (A. Diakité).

https://doi.org/10.1016/j.autcon.2018.07.025
Received 13 December 2017; Received in revised form 11 July 2018; Accepted 31 July 2018

the obstacles according to the size of the drone. Using this algorithm, the *safe shortest path* and *safe least cost path* are developed and tested. Both paths can make the drone fly under or above obstacles. The latter path allows a flight at a certain height.

This work is organized as follows. Section 2 presents the literature review. Section 3 presents the workflow. In Section 4, we introduce basic definitions and the 3D propagating approximate Euclidean distance transformation (3D PAEDT) algorithm. Section 5 analyses navigation path options for an indoor drone. Section 6 describes the methods to find SSP and SLCP using the A* and distance transformation algorithms. In Section 7, the methods are tested based on a two-story building model. Finally, conclusions are presented in Section 8.

2. Related work

3D path finding algorithms are closely related to true 3D indoor models. The present 3D models can be classified into two large groups: surface-based and volume-based models [34]. For the purpose of the 3D path finding to be able to solve the cases mentioned in Section 1, volume-based models are more appropriate. The volume-based models provide a straight-forward 3D topology between the volumetric cells (e.g. voxels, tetrahedrons), which are much simpler than the 3D topology maintained in surface-based models. Voxel and octree models are two basic 3D volume-based models.

The voxel model is used to decompose the indoor environment into a set of non-overlapping, equal-sized, simple 3D cells, called voxels. Usually these models result in a large number of units, especially if a fine resolution is required. However, the indoor environment is much smaller than the outdoor environment and therefore the universal path finding can be performed on an off-board computer. Normally, the memory requirement for a voxel model for indoor space is quite acceptable. The octree model is a compressed 3D volume model. Although the octree model can reduce memory consumption, the construction and editing of the octree is time-consuming. Due to hierarchical data structure of the octree, neighborhood operations, as for the path finding algorithm, are more time-consuming for implementation compared to voxel model. Several studies have conducted 3D indoor modeling using voxel models [15–17]; however, the path finding research performed following the modeling is limited. Wurm et al. [18] reviewed the 3D indoor modeling approach and presented a model using octree, called OctoMap, to perform collision-free navigation for mobile manipulation [19]. However, the model is only suitable for the manipulation of robots that move on the floor. Rodenberg et al. [35] and Fichtner et al. [36] have presented an octree approach for navigation of humans, but again considering a navigation on the octree cells that represent the surface where humans can walk on. Their research has shown that 'air' octree cells can become too large and the path for a drone would be rather approximate. Therefore the study presented in this paper is based on the voxel model.

Path planning based on a 2D grid or quadtree has been well-studied for mobile robots. The traditional methods based on 2D grids or quadtree models for universal path planning include A* search and distance transformation. Both approaches can be extended to 3D grids. The A* search algorithm achieves high efficiency using a heuristic function. MacAllister et al. [9] and Xu et al. [10] presented a close-to-optimal 3D path algorithm for the autonomous drone in voxel and octree models, respectively, using the A* graph search algorithm. However, the path's distance from the obstacles was not considered; thus, the path may be very close to the obstacles and unsafe for the drone.

3. Methodology

The concept of distance transformation was first proposed on a binary image by Rosenfeld and Pfaltz [20]. Distance transformation is used to convert a digital binary image that consists of object (foreground) and non-object (background) pixels into another image, in which each background pixel has a value corresponding to the minimum distance from the object by a distance function. The method is widely used in path planning because it easily supports many special navigation behaviors of moving objects [21,22]. Many algorithms on distance transformation have been proposed. Raster scanning algorithms [23–25] and propagation algorithms [26–29] are common techniques to produce distance transformation. In order to plan a safe path, Zelinsky [22] presented an "obstacle transform", a type of raster scanning distance transformation, where the obstacle cells become the targets to prevent the mobile robots from moving too close to the obstacles. However, the distance must be checked several times for all free cells; the process may be slow when it is extended to 3D. Barraquand and Latombe [30] proposed "numeric potential fields" to maximize clearance from the obstacles, but this method can move the solution path too far from the shortest path and may guide the robot through narrow free space channels.

As mentioned previously, flying drones indoors can serve many different purposes. Depending on the application, a different path may be required. In this paper, we consider two specific tasks: flying drones at a safe distance from surrounding objects and flying them at a given height above objects. The first case is important for any path computation indoors. The latter case is of specific importance for maintenance and inspection. For example, a drone can be sent to fly above pipelines mounted on the floor, or for collecting first impression of damages after an incident (e.g. fire, flood). We develop and test two algorithms that can provide:

1) a *safe shortest path* (SSP) that ensures a safe distance from indoor objects, while maintaining the shortest distance to the destination.
2) a *safe least cost path* (SLCP) that ensures a safe distance from objects and a given constant height above the floor and stairs.

Our study combines A* algorithm and distance transformation to plan such special paths for an indoor drone. Our 3D PAEDT algorithm allows to:

- create 3D buffer zone around the obstacles efficiently, in which the drone cannot fly.
- perform 3D propagating distance transformation from the obstacles to ensure the drone's safety and at the same time prevent it from flying through narrow channels.
- compute a 3D distance map from an assumed plane which has a fixed height above the floor and the stairs as an additional cost field for producing SLCP.
- combine with A* to produce 3D path efficiently.

The workflow is shown in Fig. 1. First, a 3D buffer around the obstacles is computed using 3D PAEDT, in which the distance is the

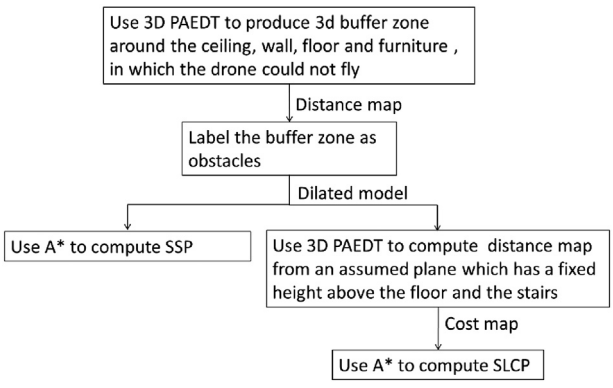

Fig. 1. Flowchart for achieving paths based on 3D PAEDT and A*.

distance from all the objects including the roof, wall, floor and furniture. Then, the 3D buffer zone is labeled as an obstacle. Thus, a new binary map made of obstacle and background is produced. It is a dilated model. Based on this dilated model, SSP can be computed (the left branch in Fig. 1). On the other hand, a distance transformation is computed again using 3D PAEDT (the right branch in Fig. 1) on the dilated model, but the distance is the distance from the free voxel to an assumed plane along which the drone flies closely and obstacles are not involved in the distance computation. The plane has a fixed height above the floor and the stairs. This distance map is regarded as a cost map. Based on it, SLCP is computed using A*. The following chapters will elaborate on these steps.

4. 3D propagation approximate Euclidean distance transformation

4.1. Basic notations and definitions

The volume-based method decomposes a 3D indoor environment into a set of non-overlapping simple cells, called voxels. We assume that the voxel V can be 26-connected, and let 26 neighbors of V be denoted by $N(V)$, p is a voxel that belongs to $N(V)$. There are three types of neighborhoods: face-touched neighborhood, edge-touched neighborhood, and point-touched neighborhood, denoted by $FN(V)$, $EN(V)$, and $PN(V)$ respectively. With the existence of obstacles, the straight distance path between any two voxels may not exist in many cases; thus, it is difficult to directly compute their Euclidean distance only using voxels' coordinates, especially in 3D space. However, the accuracy is less important than the computational performance. Therefore, in this study, an approximate Euclidean distance metric d_{ae} is computed instead of the exact Euclidean distance. The approximate Euclidean neighbor distance between V and its neighbor p is denoted by $d_{ae}(p, V)$. Considering both the computing speed and accuracy $d_{ae}(p, V) = 3$, $p \in FN(V)$; $d_{ae}(p, V) = 4$, $p \in EN(V)$; $d_{ae}(p, V) = 5$, $p \in PN(V)$ (see Fig. 2), because integer operations are computationally faster than floating-point operations and the error is small [31].

In the 3D indoor voxel model, some voxels are occupied by objects such as the walls, ceiling, floor, and furniture. Some or all of them can be aggregated and marked as O together. The other free voxels are marked as B, which represents background voxels. From the indoor voxel model that consists of O and B, the distance transformation, in a 3D configuration space, is to make a distance map, in which the value of any voxel is the approximate Euclidean distance from this voxel to the nearest voxel of O.

$$d(V) = min\{d_{ae}(V, V'), V' \in O\} \quad (1)$$

where V' is the voxel that belongs to O. For O being able to represents different objects, the results of $d(V)$ may be different. If O represents all the objects in the indoor environment, $d(V)$ represents the least distance from the voxel to the obstacle. If O only represents one object, e.g. the floor, $d(V)$ represents the least distance from the voxel to the floor.

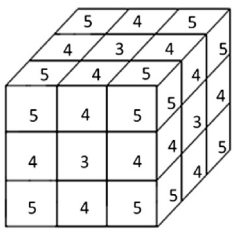

Fig. 2. Approximate Euclidean distance between neighbors.

Input: 3D indoor voxel model consisting of O and B,
Output: 3D distance map $d(V)$
Initialization:
1: Let M be an upper limit distance;
2: for $V \in O$ $\{d(V) = 0, V \rightarrow \text{bucket}(0)\}$;
3: for $V \in B$ $d(V) = M$;
4: $d = 0$;
5: for $p \in FN(V), d_{ae}(p, V) = 3$; for $p \in EN(V), d_{ae}(p, V) = 4$; for $p \in PN(V), d_{ae}(p, V) = 5$
Iterations:
6: for all V in bucket(d)
7: while (conditions according to the application)
8: { for all $p \in N(V)$
9: if $(p \in B)$ then
10: $\{newD = d(V) + d_{ae}(p, V)$;
11: if $newD < d(p)$
12: { if p has been put into bucket$(d(p))$ remove it from bucket$(d(p))$;
13: $d(p) = newD$;
14: $p \rightarrow$ bucket$(newD)$;
15: }
16: }
17: clear bucket$(d(p))$;
18: delete bucket$(d(p))$;
19: $d = d + 1$;
20: }

Fig. 3. 3D PAEDT algorithm.

4.2. 3D propagating approximate Euclidean distance transformation algorithm

In this section, we present the extended 3D PAEDT, which is the basis of our indoor path computation (Fig. 1). It is a type of width-first algorithm, similar to the one introduced by Cuisenaire and Macq. The distance metric used by Cuisenaire and Macq [26] is the Euclidean distance, whereas the distance metric used in our approach is the approximate Euclidean distance. The idea of PAEDT is to deliver the distance information iteratively via neighbor voxels in an increasing distance order. Buckets are used to store the voxels in the propagation front as they are sorted according to their indices. Since the neighbor distances used in this study are integers, the path length is also an integer. The voxel V is put into the bucket whose index is the same as its distance $d(V)$. The distance of each voxel in the propagation front from the obstacles is stored.

In contrast to the raster scanning method [23], the propagation method does not need to compute the distance for all free cells according to the application. For example, producing a 3D buffer around the obstacles in our application does not require to compute every voxel's distance. If both methods use the same size of local neighborhoods, every empty voxel should be checked the same number of times as in the process of distance transformation. For example, if the size of local neighborhoods is $3 * 3 * 3$, then every empty voxel should be checked 26 times in both methods. Thus, the propagation method will be quicker than the raster scanning method when the distances of all voxels do not have to be calculated, for example producing 3D buffer around obstacles (mentioned in Section 6.1). In the worst-case scenario, the propagation method would calculate the distance for all free voxels, similar to the raster scanning method. The algorithm is implemented as follows (Fig. 3):

The condition for the “while” iteration (line 7 in Fig. 3) determines the length of the calculation. It will be different for specific applications. The lines from 8 to 20 in Fig. 3 compose the main part of 3D PAEDT.

5. The types of the path

Having indexed all voxels according to their distance from the obstacles, we can compute paths for flying a drone. In this paper we consider three path criteria:

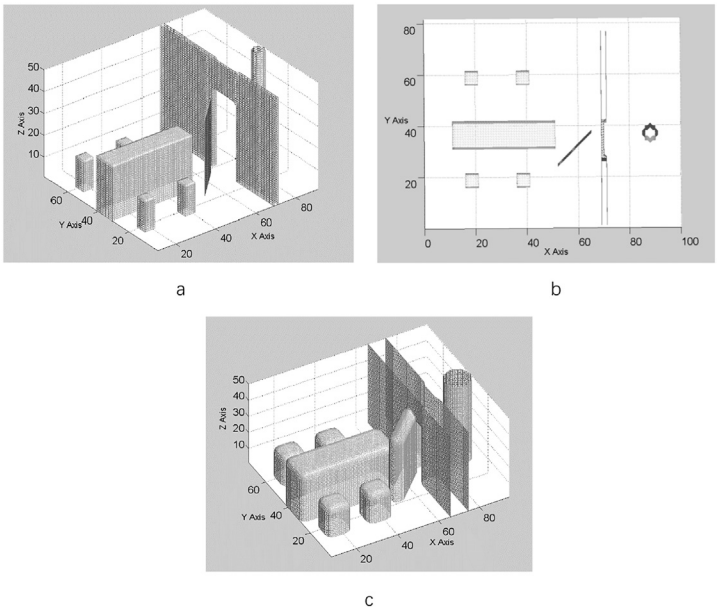

Fig. 4. The 3D buffer of the obstacles. a) Side view of the original indoor experimental data. b) Overhead view of the original indoor experimental data. c) 3D buffer around the obstacles. The buffer distance is 25 cm.

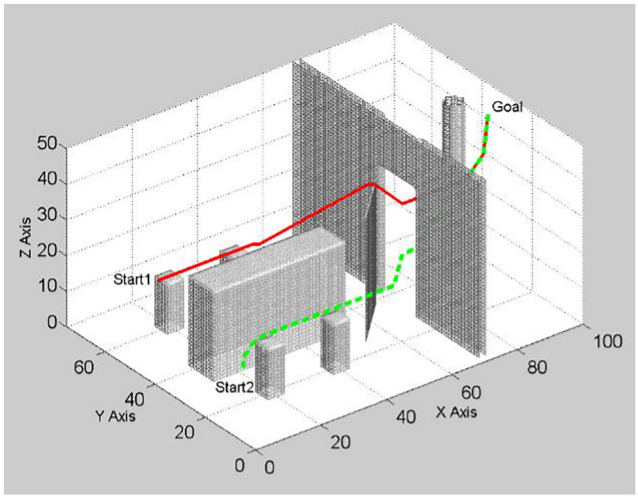

Fig. 5. The results of the safe shortest path finding. The positions of Start1 and Start2 are (10, 52, 20) and (10, 18, 10), respectively. The target position is (98, 35, 45). The red solid line represents the shortest path from a voxel indicated with Start1 to the target voxel indicated with Goal. The green dashed line represents the shortest path from Start2 to Goal. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

- Collision avoidance. In other words, the drone should not fly too close to the obstacles. Therefore, the first criterion is that the drone should fly at a minimal distance from the obstacles. Since each drone has a different size and shape, we can set a minimal distance from the obstacles based on the size of the drone. That distance should be greater than half of the width or height of the drone.
- Shortest path. The shortest path is the most common path. Depending on the considered weight parameters (e.g. distance, time) different shortest paths can be obtained. For example, when the drone delivers coffee to the customer, the path needs to be the shortest in time to ensure that the coffee is still hot.
- Height of flight. If the drone can fly at a fixed height above the floor and stairs as far as possible, it can meet fewer obstacles and save energy. Therefore in many cases, a fixed height is a basic path criterion.

As mentioned previously, we concentrate on two types of paths SSP and SLCP. The first path satisfies the first two criteria and the second considers all three criteria. Having in mind the 3D PAEDT presented in the previous section, both paths will have a minimum clearance from the obstacles first. The goal of the second path is to make the drone fly as far as possible at a fixed height h above the floor and stairs. Supposing there is a plane F, which has a fixed height h above the floor and stairs. The voxels occupied by F are marked as O_F. The cost function to move from a voxel p to V can be defined as:

$$cost = d_{ae}(p, V) + \alpha \times |d(V) - h| \quad (2)$$

where p is V's predecessor on the path; it is also one of V's neighbors, and d_{ae}(p, V) represents the approximate Euclidean neighbor distance between voxels p and V; here $d(V)$ represents the least distance from the voxel V to the floor and stairs. Therefore $|d(V) - h|$ is the least distance from V to F. It represents the cost incurred by including V on the path. For the shortest path, α is zero. The cost only depends on the actual distance travelled between p and V. For the least cost path, the cost depends both on the actual distance travelled between p and V and the distance from V to F. The greater the distance from V to F, the greater $|d(V) - h|$, and the greater the cost. If $V \in O_F$, $|d(V) - h|$ is zero and the cost is less. Therefore, the second path can make the drone fly closely along surface F, which has a fixed height h above the floor and stairs. α is a positive constant which determines by how far the resultant path will fly along surface F. Increasing the coefficient α will create a path that lets the drone fly at a fixed height closer to F. Conversely, decreasing the coefficient α leads to a shorter path. We will discuss how α influences the path based on the experiment in Section 6.

6. Indoor path finding using A* and 3D PAEDT

An experimental voxel model as shown in Fig. 4a, b, was used to evaluate our workflow (Fig. 1) and algorithm. The data contains $100 * 75 * 50 = 375{,}000$ voxels with a resolution of 5 cm. The three coordinates (x, y, and z) in Figs. 4–6 represent the indices of the voxels. The objects in the experimental data include one door, one wall, some cubic obstacles, one plate as an obstacle, and one cylindrical obstacle connecting the ceiling and the floor. There is a narrow passage between

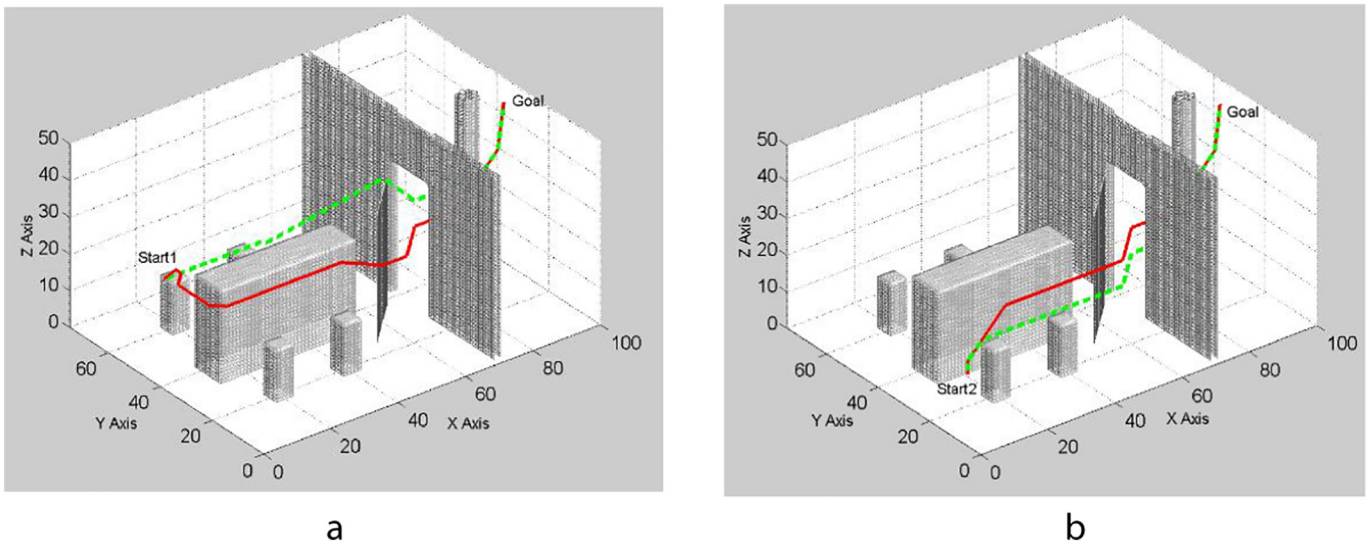

Fig. 6. Comparison of the results of SSP and SLCP. The positions of Start1 and Start2 are (10, 52, 20) and (10, 18, 10), respectively. The position of the goal is (98, 35, 45). The red line represents the least cost path and the green dashed-line represents the shortest safe path. For the least cost path in this test, h is 125 cm and α is 1. a) Comparison between SSP and the SSCP from Start1 to Goal and b) comparison between SSP and SLCP from Start2 to Goal. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

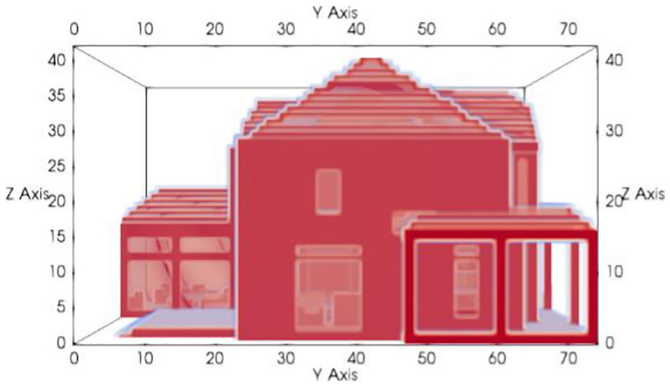

Fig. 7. 3D building model for the experiment.

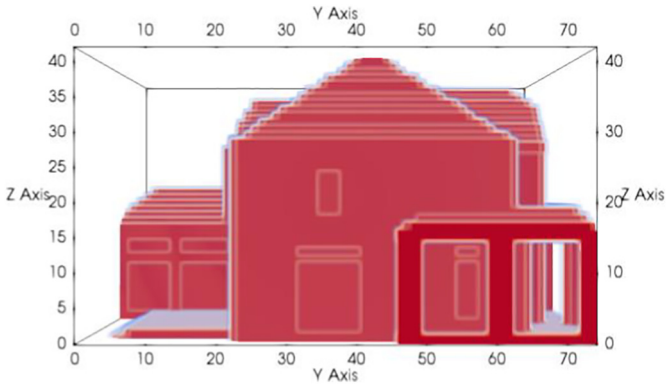

Fig. 8. Dilated 3D building model.

the door and the wall. Empty space includes 349,618 voxels. In order to visualize the inside of the room clearly, the ceiling and the wall are not shown in the figures.

6.1. The 3D buffer around obstacles using 3D PAEDT

As shown in Fig. 1, the first step is producing a 3D buffers around the objects. The buffer distance is set according to the size of the indoor drone and the application. The minimal distance of all voxels from the central objects in the buffer should be less than or equal to the buffer distance. The 3D PAEDT presented above computes the distance from the central object to a point at the edge of the buffer zone, where O in Eq. (1) represents all the objects in the room. Thus, the distances should be computed only for the voxels in the buffer zone, not for all free voxels. The 3D buffer computed using 3D PAEDT is shown in Fig. 4c.

Assuming that the same neighbor sizes (26) are used, each voxel involved in the distance calculation will be checked 26 times for both 3D PAEDT and raster scanning algorithms. However, only 79,443 empty voxels are involved in the process of producing the buffer using 3D PAEDT and all 349,618 empty voxels are involved using the raster scanning algorithm. Therefore, the 3D PAEDT algorithm is faster than the raster scanning algorithm.

After computing the 3D buffer around the obstacles, the 3D buffer zone is labeled as obstacles and 3D indoor model is dilated. Then SSP and SLCP are computed on the dilated indoor model.

6.2. A* algorithm

In order to plan a shortest path from the start to the goal on the voxel model, the well-known A* algorithm can be defined as the evaluation function f of a voxel V:

$$f(V) = g(V) + h(V) \tag{3}$$

where $g(V)$ is the cost of the path from the start to V, and $h(V)$ is the heuristic estimate of the cost of the remaining path from V to the goal. It is calculated as the Euclidean distance between V and the goal in this study. According to the cost function (Eq. (2)), we define $g(V)$ as

$$g(V) = g(p) + d_{ae}(p, V) + \alpha \times |d(V) - h| \tag{4}$$

where V is the last free voxel on the path, $g(p)$ is the cost of the path from the start voxel to V's predecessor, p, on the path. For the shortest path, α is zero. For the least cost path, α is a positive constant.

6.3. Safe shortest path finding

In the dilated mode produced by 3D PAEDT, two voxels are selected one as target and one as the start. SSP can be calculated using the A* algorithm (Eq. (4)). Fig. 5 shows two shortest paths with the different starts and the same target. They are planed on the dilated model above,

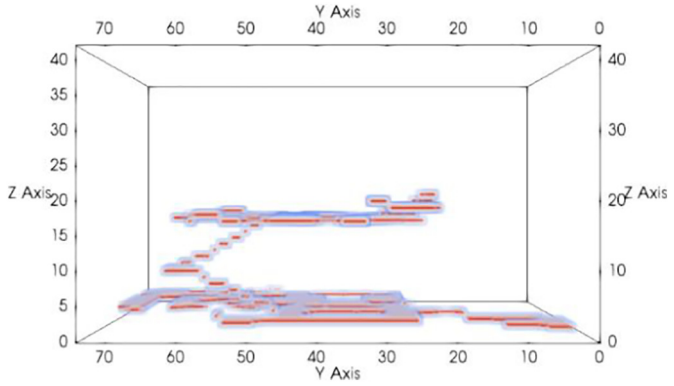

Fig. 9. Floor and stairs.

therefore they have a minimal distance of 25 cm from the obstacles and do not go through the narrow passages between the door and the wall; thus, they are safe for the flight of the drone.

6.4. Safe least cost path finding

The second intended path is the least cost path between the Start and Target voxels on the dilated model produced previously. The experimental indoor model does not include stairs and contains only a horizontal floor. Suppose there is a plane F, whose height from the floor is h. The cost of the path is the sum of the actual distance travelled along the path from the start to the target and the distance from the path to F according to Eq. (2). Therefore, $d(V)$ in Eqs. (2) and (4) is the distance from V to the floor for this test. The greater the distance to F, the greater the cost. Therefore, the drone can fly close to surface F. First, the distance from every free V to F should be computed using 3D PAEDT. Then SLCP is produced using A*. SLCP lets the drone fly at a fixed height h as far as possible when the floor is horizontal compared to SSP (Fig. 6). The method can also be applied to more complex indoor models that include stairs and furniture (see Section 6).

7. Experiment

The method presented in Sections 4 and 5, is tested on a two-story building model with stairs and furniture (Fig. 7). It is a voxel model with a resolution of 20 cm. Three coordinates (x, y, and z) in Figs. 7 to 12 represent the indices of the voxels. The model contains 222,524 voxels. The empty space contains 178,310 voxels. This model originated from the 3D Warehouse model (https://3dwarehouse.sketchup.com/model/caf16501182de5cca59adc119bf6af9/country-home-w-furnished-interior), which is a boundary representation model designed by Sketchup and it is voxelized using binvox (http://www.patrickmin.com/binvox/).

Initially, the floor, furniture, walls, ceiling, and stairs are all regarded as obstacles with the same values (1), i.e. no distinction is made between different objects. All other voxels are considered free voxels. Fig. 8 shows the 3D buffer around the obstacles calculated using 3D PAEDT; the buffer distance is 40 cm and the obstacles are dilated. After this step only 24,951 empty voxels are involved in the process of producing a buffer using 3D PAEDT and all 178,310 empty voxels are involved using the raster scanning algorithm. Thus, 3D PAEDT is faster than the raster scanning algorithm.

The algorithm is also tested on the same building model but with a 10-cm resolution. Table 1 shows the comparison of the computation time and processed voxels between 3D PAEDT and raster scanning with a different dilation radius. For 3D PAEDT, the computation time and processed voxels are related to the dilation radius. The smaller the dilation radius, the fewer voxels are processed and the faster the computation is. For raster scanning, processed voxels are as many as the free voxels; thus, they are not related to the dilation radius. The computation speed of both algorithms depends on the model resolution. For the model resolution of 20 cm and the dilation radius of 20 cm, the 3D PAEDT algorithm is the fastest.

Next, horizontal surfaces are extracted from the model by selecting all non-empty space voxels that have an empty space voxel above them. Then the floor and the stairs are assigned to the same segment from the horizontal surfaces using a flood-fill algorithm [32], which starts at the voxel with the lowest elevation and then expands in all directions, adjacent voxels are assigned to the same segment whereas disconnected voxels are assigned to new segments. The flood-fill algorithm is capable of expanding upwards and downwards by a given predefined number (vertical foot span). Fig. 9 shows the floor and the stairs extracted from the obstacles.

Next, the distance field is computed using 3D PAEDT. Assuming that there is a plane (not a flat surface) whose distance from the floor and the stairs is 100 cm, we will plan a path that lets the drone fly along this plane closely. The distance $d(V)$ from the floor and stairs for free voxels is calculated first. Then, the distance from the plane can be obtained using the formula $|d(V) - 100|$. The distance field represents the distance to the plane. Fig. 10 shows the slices of distance images.

Finally, the safe shortest path can be planned based on the dilated building model using the A* algorithm. The least cost path can be planned based on the dilated model and the distance field (Fig. 10b). Fig. 11 shows the path planning results. The safe shortest path tends to go towards the ceiling, but it keeps a minimal distance from the ceiling (40 cm). The least cost path lets the drone fly at a fixed height of 100 cm from the floor and stairs, in which α in Eq. (4) is 1.

If h and α are changed, the paths will be different. Fig. 12 shows the different paths when h and α are changed. Table 2 shows the lengths of the path with the same h(100 cm) and different weight α. It shows that

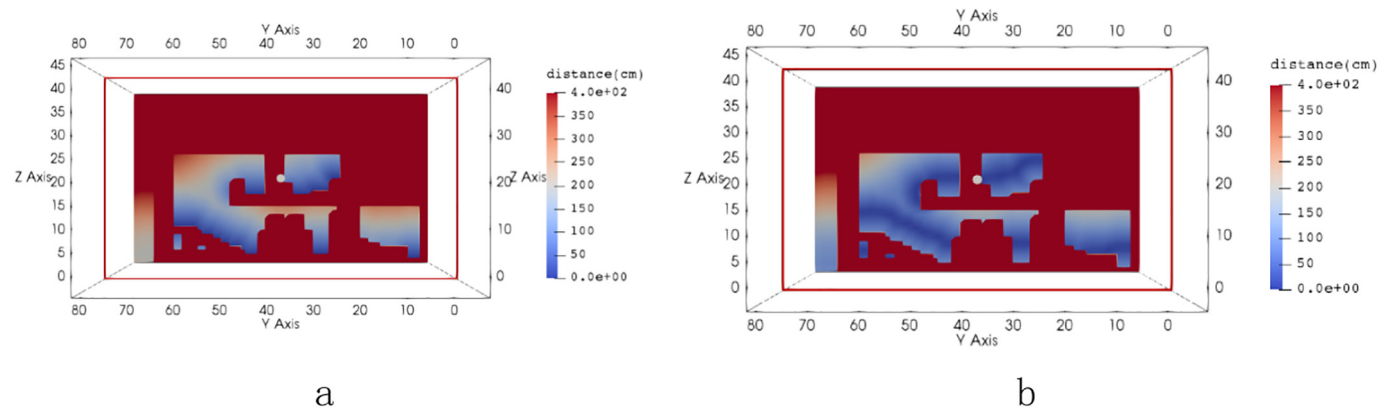

Fig. 10. Slices of distance images. a) Distance from the floor and the stairs, b) distance from the plane whose distance from the floor and the stairs is 100 cm.

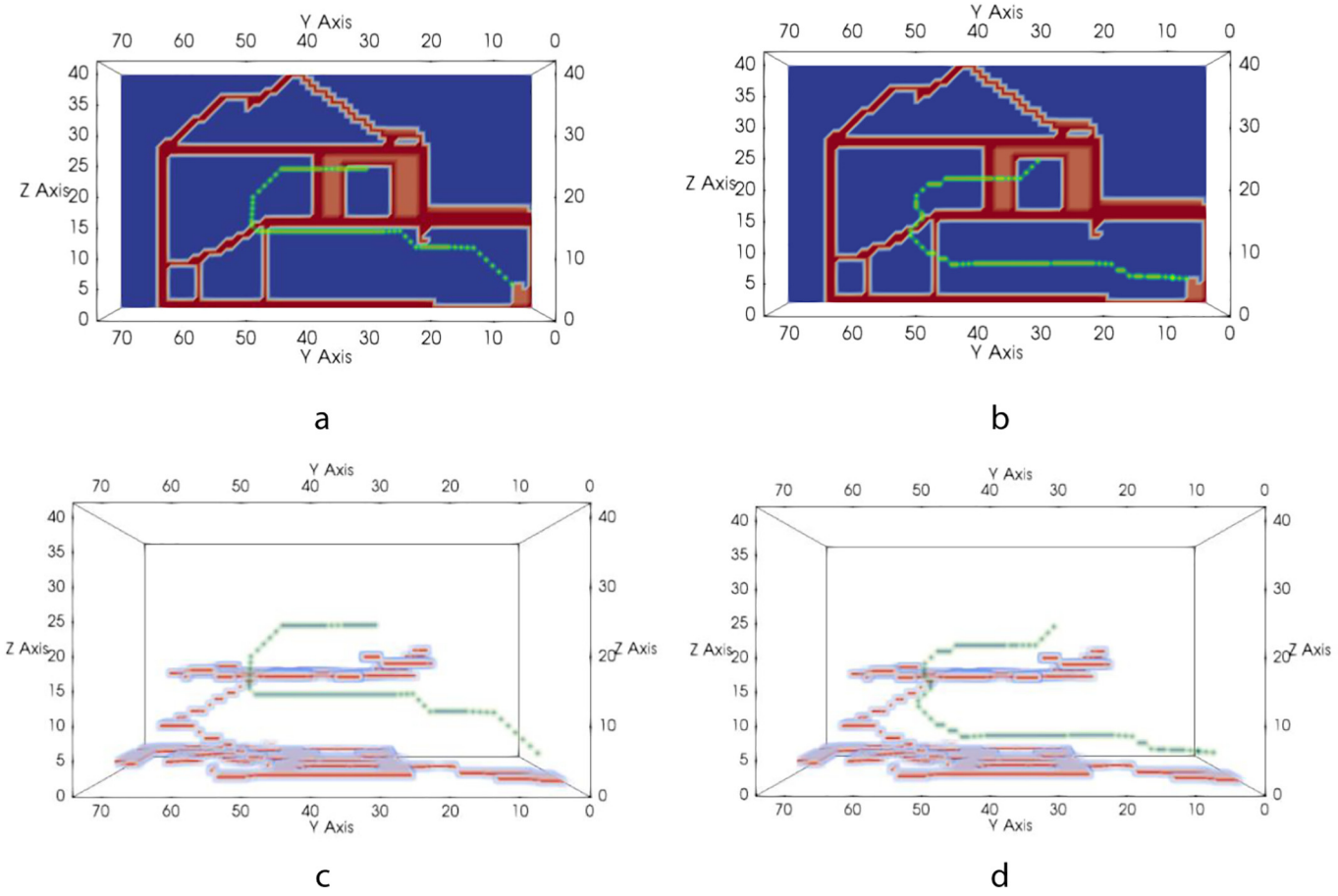

Fig. 11. Path planning results. The position of the start is (15, 5, 5) located on the first floor. The position of the end is (20, 30, 25) located on the second floor. a) SSP with a slice of the building model. b) SLCP with the slice of the building model. c) SSP with stairs and the floor. d) SLCP with stairs and the floor.

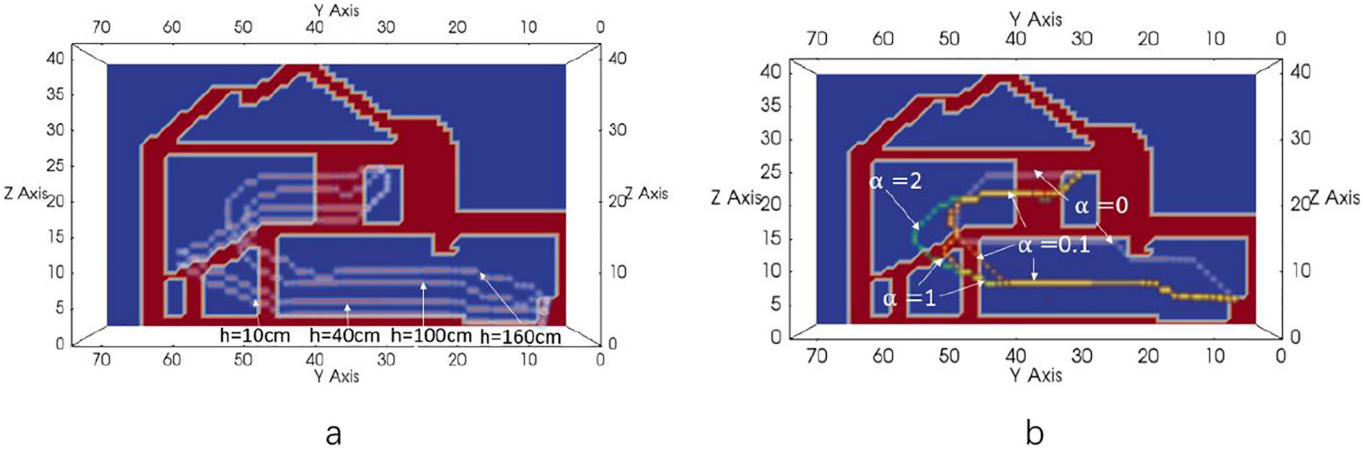

Fig. 12. Different paths with the slice of the dilated building model whose resolution is 20 cm. a) SLCPs with the same α (1) and different h values (10 cm, 40 cm, 100 cm, and 160 cm); the higher the value of *h*, the greater the distance of the path from the floor and stairs is. b) The paths with different α(0, 1, 2); $h = 100$ cm when $\alpha = 1$ and 2. When α is 0, the path is the safe shortest path. The greater the α value, the longer the path, and the closer to the plane *F* whose distance from the floor and stairs is 100 cm, especially when the path goes up along the stairs.

the greater the value of α, the greater the length. However when α is greater than 2, the length of the path changes slightly or does not change. Fig. 12(b) shows that the paths are almost the same when the floor is flat, and different when the path goes up along the stairs.

8. Conclusions and future work

This paper presents an easy and efficient approach to compute 3D universal safe paths for drones in a known environment using a voxel model. We develop and test the 3D PAEDT algorithm, which is the core of the presented work. The algorithm allows to reduce the number of voxels to be considered for path computation by filtering all the obstacles and considering the size of the drone. This is to say that the minimal distance between the obstacles and the drone can be set according to the size of the drone. This approach prevents drones from flying too close to the obstacles and into narrow passages. The 3D

Table 1
Computation time and processed voxel comparison between 3D PAEDT and raster scanning.

Resolution (cm)	Number of voxels	Dilation radius (cm)	3D PAEDT		Raster scanning	
			Voxels being processed	Computation time (s)	Voxels being processed	Computation time (s)
20	222,524	20	42,588	0.933	178,310	72
		30	74,191	1.779		
		40	76,525	2.469		
10	1,780,199	10	249,787	126.278	1,587,114	587
		20	425,517	180.61		
		30	574,397	235.54		
		40	704,220	310.734		

Table 2
Length of the path with different weights α and same *h*(100 cm).

α	0	0.1	0.5	1	2	3	4	5
The length of the path (m)	14	14	14.6	15.2	16.6	16.8	16.8	17.2

PAEDT algorithm does not need to calculate the distance for all free voxels in most cases; thus, it is more efficient than for example the raster scanning distance transformation algorithm. The computation time of 3D PAEDT is related to both the model resolution and dilation radius. It should be greater than half of the width or height of the drone. The smaller the drone, the smaller dilation radius. However, the voxel size should be less than or equal to the dilation radius. If the voxel size is too big, 3D buffers may create some narrow passages that seem unnavigable, but in fact, the passage may be safe for the drone to fly through. In other words, the smaller the voxel size, the more accurate the 3D buffer. At the same time, the higher the voxel model resolution, the greater the time complexity of 3D PAEDT. Therefore, there should be a tradeoff between the buffer accuracy and time complexity when the voxel size is set, as a finer buffer will imply a higher resolution.

Two types of universal paths are presented based on the 3D PAEDT: the safe shortest path and the safe least cost path. The experiments show that both types of paths are safe regardless of the shape and position of the obstacles.

Although only two types of paths are presented in this study, the methods can be extended to compute other types of paths by changing the cost function in Eq. (2). The experiments in this research only labeled the floors, stairs, and obstacles and do not consider other features of the building, such as the number and function of the rooms; however, with the development of 3D indoor model with rich semantic [33], the methods can be extended to plan context-aware smart paths by transferring the semantic to navigable space labels depending on the application.

This study focuses entirely on 3D path planning algorithms for drones in the indoor environment, which is static and completely known. In the future, we will extend the algorithm to plan 3D collision-free paths for drones when the indoor environment is not static or partially known, i.e., when there are moving obstacles in the room.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 41472113) and the Dutch STW/M4S program within the project Sims3D (No. 13742, www.sims3d.net).

References

[1] A. Hussein, A. Al-Kaff, A. de la Escalera, J.M. Armingol, Autonomous indoor navigation of low-cost quadcopters, IEEE International Conference Service Operations and Logistics, and Informatics (SOLI), 2015, pp. 133–138, , https://doi.org/10.1109/SOLI.2015.7367607.

[2] S. Grzonka, G. Grisetti, W. Burgard, A fully autonomous indoor quadrotor, IEEE Trans. Robot. 28 (1) (2012) 90–100, https://doi.org/10.1109/TRO.2011.2162999.

[3] Y. Ham, K.K. Han, J.J. Lin, M. Golparvar-Fard, Visual monitoring of civil infrastructure systems via camera-equipped unmanned aerial vehicles (UAVs): a review of related works, Vis. Eng. 4 (1) (2016) 1, https://doi.org/10.1186/s40327-015-0029-z.

[4] P. Liu, A.Y. Chen, Y.N. Huang, J.Y. Han, J.S. Lai, S.C. Kang, T.-H.R. Wu, M.-C. Wen, M.-H. Tsai, A review of rotorcraft unmanned aerial vehicle (UAV) developments and applications in civil engineering, Smart Struct. Syst. 13 (6) (2014) 1065–1094, https://doi.org/10.12989/sss.2014.13.6.1065.

[5] M.D. Phung, H.Q. Cong, T.H. Dinh, Q. Ha, Enhanced discrete particle swarm optimization path planning for UAV vision-based surface inspection, Autom. Constr. 81 (2017) 25–33, https://doi.org/10.1016/j.autcon.2017.04.013.

[6] A.G. Bachrach, R. He, N. Roy, Autonomous flight in unknown indoor environments, Int. J. Micro Air Veh. 1 (4) (2009) 217–228, https://doi.org/10.1260/175682909790291492.

[7] A. Bry, C. Richter, A. Bachrach, N. Roy, Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments, Int. J. Robot. Res. 34 (7) (2015) 969–1002, https://doi.org/10.1177/0278364914558129.

[8] R. He, Planning in information space for a quadrotor helicopter in a GPS-denied environment, IEEE International Conference on Robotics & Automation, 2008, pp. 1814–1820, , https://doi.org/10.1109/ROBOT.2008.4543471.

[9] B. MacAllister, J. Butzke, A. Kushleyev, H. Pandey, M. Likhachev, Path planning for non-circular micro aerial vehicles in constrained environments, IEEE International Conference Robotics and Automation (ICRA), 2013, pp. 3933–3940, , https://doi.org/10.1109/ICRA.2013.6631131.

[10] S. Xu, D. Honegger, M. Pollefeys, L. Heng, Real-time 3D navigation for autonomous vision-guided MAVs, IEEE/RSJ International Conference Intelligent Robots and Systems (IROS), 2015, pp. 53–59, , https://doi.org/10.1109/IROS.2015.7353354.

[11] A.A. Diakité, S. Zlatanova, First experiments with the tango tablet for indoor scanning, ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, III-4, 2016, pp. 67–72, , https://doi.org/10.5194/isprs-annals-III-4-67-2016.

[12] L. Díazvilariño, P. Boguslawski, K. Khoshelham, et al., Indoor navigation from point clouds: 3D modelling and obstacle detection, ISPRS International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLI-B4, 2016, pp. 275–281, , https://doi.org/10.5194/isprs-archives-XLI-B4-275-2016.

[13] L. Liu, S. Zlatanova, A two-level path-finding strategy for indoor navigation, Intelligent Systems for Crisis Management, Springer, Berlin Heidelberg, 2013, pp. 31–42, , https://doi.org/10.1007/978-3-642-33218-0.

[14] Q. Xiong, Q. Zhu, S. Zlatanova, Z. Du, Y. Zhang, L. Zeng, Multi-level indoor path planning method, ISPRS International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XL-4/W5, 2015, pp. 19–23, , https://doi.org/10.5194/isprsarchives-XL-4-W5-19-2015.

[15] H. Moravec, Robot spatial perception by stereoscopic vision and 3D evidence grids, Tech. Report, Robotics Institute, Carnegie Mellon University, September, 1996 CMU-RI-TR-96-34.

[16] Y. Roth-Tabak, R. Jain, Building an environment model using depth information, Computer 22 (6) (1989) 85–90, https://doi.org/10.1109/2.30724.

[17] N. Duffy, D. Allan, J.T. Herd, Real-time collision avoidance system for multiple robots operating in shared work-space, IEE Proc. E Comput. Digit. Tech. 136 (6) (1989) 478–484, https://doi.org/10.1049/ip-e.1989.0065.

[18] K.M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, W. Burgard, OctoMap: a probabilistic, flexible, and compact 3D map representation for robotic systems, Proceedings of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation, vol. 2, 2010 https://www.mendeley.com/research-papers/octomap-probabilistic-flexible-compact-3d-map-representation-robotic-systems/.

[19] A. Hornung, M. Phillips, E.G. Jones, M. Bennewitz, M. Likhachev, S. Chitta, Navigation in three-dimensional cluttered environments for mobile manipulation, IEEE International Conference on Robotics and Automation (ICRA), 2012, pp. 423–429, , https://doi.org/10.1109/ICRA.2012.6225029.

[20] A. Rosenfeld, J.L. Pfaltz, Distance functions on digital pictures, Pattern Recogn. 1 (1) (1968) 33–61, https://doi.org/10.1016/0031-3203(68)90013-7.

[21] R. Jarvis, Robot path planning: complexity, flexibility and application scope, Proceedings of the 2006 International Symposium on Practical Cognitive Agents and Robots, ACM, 2006, pp. 3–14, , https://doi.org/10.1145/1232425.1232430.

[22] A. Zelinsky, Environment Exploration and Path Planning Algorithms for Mobile Robot Navigation Using Sonar (Ph.D. Dissertation), University of Wollongong, Department of Computer Science, 1991, http://ro.uow.edu.au/theses/1295/.

[23] G. Borgefors, Distance transformations in digital images, Comput. Vis. Graph. Image

Process. 34 (3) (1986) 344–371, https://doi.org/10.1016/S0734-189X(86)80047-0.
[24] J.C. Mullikin, The vector distance transform in two and three dimensions, CVGIP Graph. Models Image Process. 54 (6) (1992) 526–535, https://doi.org/10.1016/1049-9652(92)90072-6.
[25] T. Saito, J.I. Toriwaki, New algorithms for Euclidean distance transformation of an n-dimensional digitized picture with applications, Pattern Recogn. 27 (11) (1994) 1551–1565, https://doi.org/10.1016/0031-3203(94)90133-3.
[26] O. Cuisenaire, B. Macq, Fast Euclidean distance transformation by propagation using multiple neighborhoods, Comput. Vis. Image Underst. 76 (2) (1999) 163–172, https://doi.org/10.1006/cviu.1999.0783.
[27] H. Eggers, Two fast Euclidean distance transformations in Z^2 based on sufficient propagation, Comput. Vis. Image Underst. 69 (1) (1998) 106–116, https://doi.org/10.1006/cviu.1997.0596.
[28] I. Ragnemalm, Neighborhoods for distance transformations using ordered propagation, CVGIP Image Underst. 56 (3) (1992) 399–409, https://doi.org/10.1016/1049-9660(92)90050-D.
[29] L. Vincent, Exact Euclidean distance function by chain propagations, IEEE Computer Society Conference on Computer Vision and Pattern Recognition Proceedings CVPR'91, 1991, pp. 520–525, , https://doi.org/10.1109/CVPR.1991.139746.
[30] J. Barraquand, J.C. Latombe, Robot motion planning: a distributed representation approach, Int. J. Robot. Res. 10 (1991) 628–649, https://doi.org/10.1177/027836499101000604.
[31] B.J.H. Verwer, P.W. Verbeek, S.T. Dekker, An efficient uniform cost algorithm applied to distance transforms, IEEE Trans. Pattern Anal. Mach. Intell. 4 (1989) 425–429, https://doi.org/10.1109/34.19041.
[32] M. Koopman, 3D Path-finding in a Voxelized Model of Indoor Environments (Master thesis), Delft University of Technology, Faculty of Architecture and The Built Environment, 2016, https://repository.tudelft.nl/islandora/object/uuid:13788271-e19d-41e1-b827-fe7535a66281?collection=education.
[33] A.A. Diakité, S. Zlatanova, Spatial subdivision of complex indoor environments for 3D indoor navigation, Int. J. Geogr. Inf. Sci. 32 (2) (2017) 213–235, https://doi.org/10.1080/13658816.2017.1376066.
[34] S. Zlatanova, Representations: 3-D, in: Richardson, Castree, GoodChild, Kobayashi, Liu, Marston (Eds.), The International Encyclopedia of Geography: People, the Earth, Environment and Technology, Wiley-Blackwell, 978-0-470-65963-2, 2017, pp. 1–17, , https://doi.org/10.1002/9781118786352.wbieg1157.
[35] O.B.P.M. Rodenberg, E. Verbree, S. Zlatanova, Indoor A* pathfinding through and octree representation of a point cloud, ISPRS Ann. Photogramm. Remote Sensing Spatial Information Sciences, IV-2/W1, 2016, pp. 249–255, , https://doi.org/10.5194/isprs-annals-IV-2-W1-249-2016.
[36] F.W. Fichtner, A.A. Diakité, S. Zlatanova, R. Voûte, Semantic enrichment of octree structured point clouds for multi-story 3D pathfinding, Trans. GIS 22 (1) (2018) 233–248, https://doi.org/10.1111/tgis.12308.