International Journal of Robotics and Automation (IJRA)
Vol.8, No.4, December 2019, pp. 277~292
ISSN: 2089-4856, DOI: 10.11591/ijra.v8i4.pp277-292

Environment mapping, map constructing, and path planning for underwater navigation of a low-cost µAUV in a cluttered nuclear storage pond

Yibin Peng, Peter N Green
School of Electrical and Electronics Engineering, the University of Manchester, UK

Article Info

Article history:

Received Apr 26, 2019
Revised Aug 27, 2019
Accepted Oct 6, 2019

Keywords:

A* path planning algorithm
Aerial mapping grid map
reconstruction and interpolation
Autonomous underwater vehicle
Map gridding
Point cloud map

ABSTRACT

This paper presents a novel approach that enables a low-cost µAUV (micro autonomous underwater vehicle) navigate and work safely in an enclosed cluttered underwater environment. In order to achieve a autonomy collision-free navigation in underwater, it requires a reliable approach that would allow a µAUV which equipped with sparse and inaccurate sonar sensors to map the underwater environment, a map constructing algorithm that converts the obtained data to useful information to establish a map, and a path planning algorithm that plans a collision-free path from the start to the desired goal. The proposed approach will integrate environment survey, environment reconstruction, and path planning as a completed task, which involves map acquisition and path planning. A complete simulation of the environment topology acquisition and route planning process is documented in this paper.

Corresponding Author:

Yibin Peng,
School of Electrical and Electronics Engineering,
The University of Manchester,
Oxford Rd, Manchester M13 9PL, Great Manchester, UK.
Email: yibin.peng@manchester.ac.uk

1. INTRODUCTION

Investigations on the autonomous vehicle have been made significant progress over the past few years. Based on its working environment, autonomous vehicles can be categorised into three types, the one working in the air is called unmanned aerial vehicles (UAVs) such as drones, which is used to film, environmental surveillance, geography study, and even parcel delivery. The one working in the ground is called unmanned ground vehicle (UGV) such as self-driving cars, which is used to replace human pilot. Moreover, the one working underwater is called autonomous underwater vehicle (AUV). This paper interests at AUV underwater application.

A diversity of AUVs has already been developed in different purposes and deployed in different underwater situations, which including industrial processes, marine research, and air crash investigations. Those industrial processes include but are not limited to undersea oil pipeline leakage detection [1] and spent nuclear fuel storage pond monitoring [2]. The ocean researches include undersea oil exploitation, marine study and information collection, ocean search and rescue [3]. As for air crash investigations, AUVs have been used to find wreckages of missing aeroplanes, such as Malaysia Airlines Flight 370 accident [4].

This work is motivated by the nuclear storage pond decommissioning problem. Storage ponds are used to keep spent fuel assemblies and other radioactive material underwater until they are sufficiently benign to be subjected to processing for long term safe storage. In terms of size, Olympic swimming pools provide a convenient benchmark, although storage ponds are typically much deeper. Waste objects are placed

on the bottom of the pond with a significant depth of clear water above for shielding purposes. The material remains within a pond for a number of years. The spent nuclear fuel storage pond contains highly radioactive metals such as Uranium and Thorium [5], which are hazardous to human health. Places such as these are nearly impossible for humans to work in even wearing a protective suit. In such circumstances, AUVs could potentially be used to replace humans to monitor for leaking containers, to find the disposition of material in older ponds, and to explore the unknown underwater environment.

Based on the era of construction, existing storage ponds can be classified as legacy ponds or modern ponds. The latter tend to be equipped with regular storage racks designed to hold canisters containing the radioactive material. Hence the distribution of clutter (the solid objects within the pond) is regular or at least well-organised [2]. Figure 1 is an example of a modern pond. Legacy ponds are a different matter. Originating from times when the hazards of storing nuclear material were not as clearly understood as they are now, and handling technology was less well developed, legacy ponds contain material that was originally stored in a haphazard manner in skips (dumpsters). Some material in legacy ponds has been immersed for many years and has partially disintegrated [2]. Figure 2 is an example of the legacy pond.

Figure 1. Modern nuclear storage pond [6]

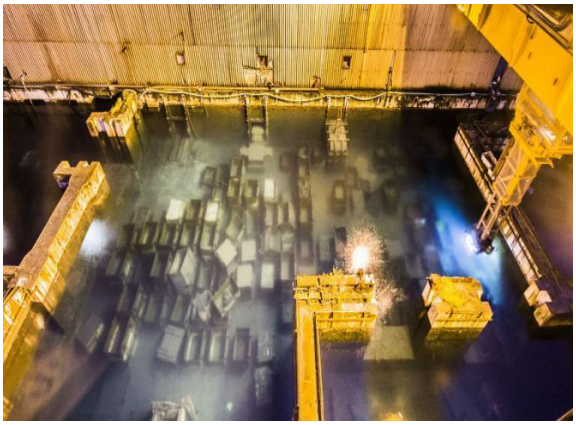

Figure 2. Legacy nuclear storage pond [7]

It is evidently desirable that conditions in both classes of ponds be monitored, although this is a potentially hazardous endeavour. In the case of modern ponds, it is helpful to check for leakage from canisters so that remedial work can be carried out in a timely manner. For legacy ponds, it is necessary to determine the dispersal of objects within a pond and the distributions of radioactivity, temperature and other parameters throughout the pond, so that plans for material removal and long-term storage can be formulated.

The research reported in this paper builds on work on small-scale, low-cost AUVs (µAUVs) that has been underway in Manchester for several years [8, 9], resulting in the AVEXIS vehicle [8, 9]. AVEXIS has 4 degrees of freedom. Supporting communication and localisation systems are currently under development.

One aim of the AVEXIS work is to develop a swarm ('shoal') of AUVs able to explore, map and monitor storage ponds. Such a system requires mapping and path planning capabilities and this paper discusses initial efforts to provide this kind of functionality.

The problem of the µAUV navigation in a storage pond is how to enables a µAUV traverse from the start position to the goal position (the destination where the robot heading to) while avoiding any obstacles in its environment without the help of GPS (GPS works badly in underwater) [10]. [11-13] introduce several navigation strategies for mobile robots. 3D path planning in a cluttered pond is a particularly interesting challenge since it relies on 3D maps whose accuracy may not be high, and collisions between robots and clutter may have serious consequences for the robot, the environment, bystanders, or all three. Moreover, the possible approaches are constrained by size and cost issues. Clearly, the µAUVs need to be small enough to navigate through confined spaces, and a key objective of this work has been to keep costs low to enable the system to be used in a variety of water-based processes.

However, so far, most approaches such as Forward-Looking Sonar [14], Multi-Beam Sonar [15] has been conducted in expensive big-sized AUV for marine research. This paper proposes an approach to address the pond mapping and path planning problem for a sensor-limited low-cost µAUV, which consists of 3 steps. Step 1, a low-cost AUV with sparse and inaccurate sonar sensors to map the underwater environment and gather the depth measurement data. Step 2, a map construction that converts the obtained data to useful information to establish an occupancy grid. Step 3, A* path-planning algorithm that plans a collision-free path from the start to the desired goal.

The environment mapping strategy is based on the idea of an 'aerial survey'. In this context, an aerial survey is simply a set of depth measurements taken on a regular measurement grid at a fixed depth within the clear water region of the ponds. See Figure 3. The result of a survey is a matrix of tuples, whose components are measurement position and corresponding depth. This data is the 2D projection of the 3D clutter on to the measurement plane and can be used to construct a height map of the objects at the bottom of the pond. This in itself provides useful data about the disposition of material within a pond. However, a significant amount of information is lost and height maps do not provide adequate support for the path planning needed for detailed exploration. What is really needed is a 3D representation of the pond clutter.

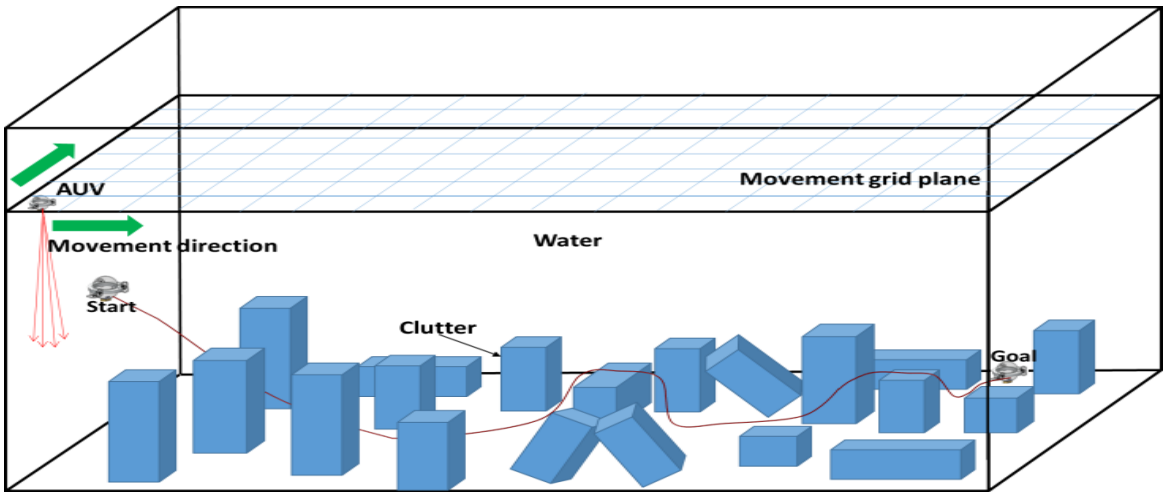

Figure 3. A complete aerial mapping and path planning in a cluttered enclosure

In this paper, the advantage is taken of the fact that most clutter in storage ponds is prismatic in shape i.e. objects have a polygonal base and a cross-section that does not vary with height (cuboid and cylindrical containers that are typically used in most ponds). The approach outlined in this paper locates clutter from a heightmap and uses this information to generate 3D point cloud map of prismatic objects from which a full 3D grid map of the pond can be constructed that is suitable for A* path planning algorithms.

In order to ensure that the A* path planning algorithm works with the obtained 3D grid map, both processes must be performed in the same coordinate system and the depth measurement at reflection positions must lie on the user-specified survey grid. The survey grid is the location of mesh points in the measurement plane where depth measurements are taken (as indicated in Figure 1) and the A* method is applied to a workspace based on a discretized representation of the environment's geometry. In this system, the A* searching grid is a 3D grid based on the 2D survey grid but has the same interval in the z-axis.

Because there is often unlikely that the sampled locations will coincide with the survey grid, such as the case that inclined objects cause the depth measurement at positions do not lie on the survey grid. Therefore, it requires further data processing to make them consistency. This paper explains how to establish a 3D digital grid map based on raw depth measurement data and provides some simulation results.

2. AERIAL MAPPING

The term of "Aerial mapping" or "aerial survey" originally come from an airborne land survey, which is an efficient method of mapping geographic feature of large areas. It is also known as Echo-sounding sonar in the undersea image and investigation research. Side-scan uses sonar devices that produce a conical or fan-shaped beam down to the seafloor with a wide beam that perpendicular to the path of the sensor through the water to gather the geographic information of the seabed [16]. Echo-sounding uses sonar devices that produce a conical beam downward to the pond bottom with a wide beam angle perpendicular to the path of the sensor through the water to measure the depth of the bottom [17].

This mapping algorithm is simply a set of depth measurements taken on a regular measurement grid at a fixed depth within the clear water region of the ponds. In practice, µAUV will project a vertical beam from a plane with a specified height toward an area by high-frequency acoustic pulse emission in order to capture depth information about the corresponding bottom surface. The movement plane is set above clutters and sampled in a uniformly spaced 2d grid (see Figure 1). The µAUV will move to each grid and measure the depth of the current grid by the received pulse that reflected off from the object's top surface. A routine of this survey is shown in Figure 4. The result of an aerial survey is an array of tuples, whose components are measurement positions and its corresponding height. See Figure 5 for the input and output of this method.

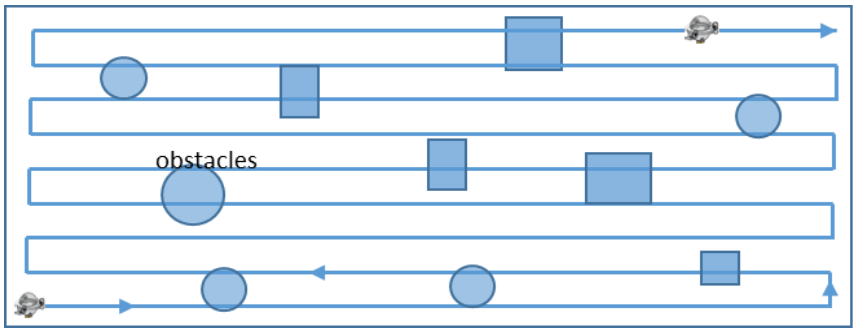

Figure 4. Scan itinerary

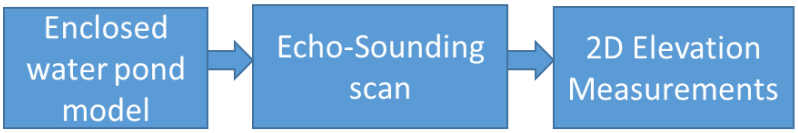

Figure 5. Diagram of input and output

The Echo-sounding is carried out by acoustic sensors which rely on modulating surface acoustic waves to sense a physical object. These devices transduce electrical signals to acoustic waves, receive reflected waves, and convert received waves back to electrical signal [18]. During the wave propagation, acoustic waves will be reflected, refracted, or attenuated by the medium. For the depth measurement, the reflection is the main concern. In terms of analysis, the form of a wave in-depth measurement is not convenient, but the characteristic of the acoustic wave can be approximated to a ray and simply neglected its wave nature since the wavelength of the typical acoustic wave is smaller compared to the size of physical objects under the water [19]. As a consequence, the ray tracing algorithm is introduced to model and simulate the sound propagation [20]. The propagation of the acoustic wave is regarded as rays, and the conical beam is constituted of a finite number of rays. The depth can be calculated by knowing the time interval between the transmitted signal and received signal and the propagation speed of the acoustic ray.

The main problem is finding the correct signal (ray) from the received signal spectrum. Because there are unwanted signals and background noises in the received signal spectrum. The background noise includes reflections from small objects, reflections of sidelobes or part of reverberation (especially in shallow waters), the unwanted signal includes later reflections (a signal that has multiple times of reflections) and diffuse reflections (a signal that scatters in all direction when hits an object). Therefore, the useful signal is the normal reflection of the emitted ray, because a receiver and a transmitter combined in one device (transceiver), so the only first-order specular reflection can be received is the normal reflection. Theoretically, it is reflected off the nearest obstacles and has the highest remaining energy [18]. Based on this property, 2 key factors can be used to estimate the normal reflection: time of flight (TOF) and impulse response energy level. In the impulse response in the time domain, the normal reflection signal is the pulse with the highest energy response and the shortest TOF. Once the TOF of the desired signal is known, the depth is given by the equation:

$$Depth = v * \frac{t}{2} \quad (1)$$

where v is the speed of sound in water and t is the time of fight of the normal reflection signal. The flowchart of the implemented ray-tracing based aerial mapping MATLAB simulator is shown in Figure 6.

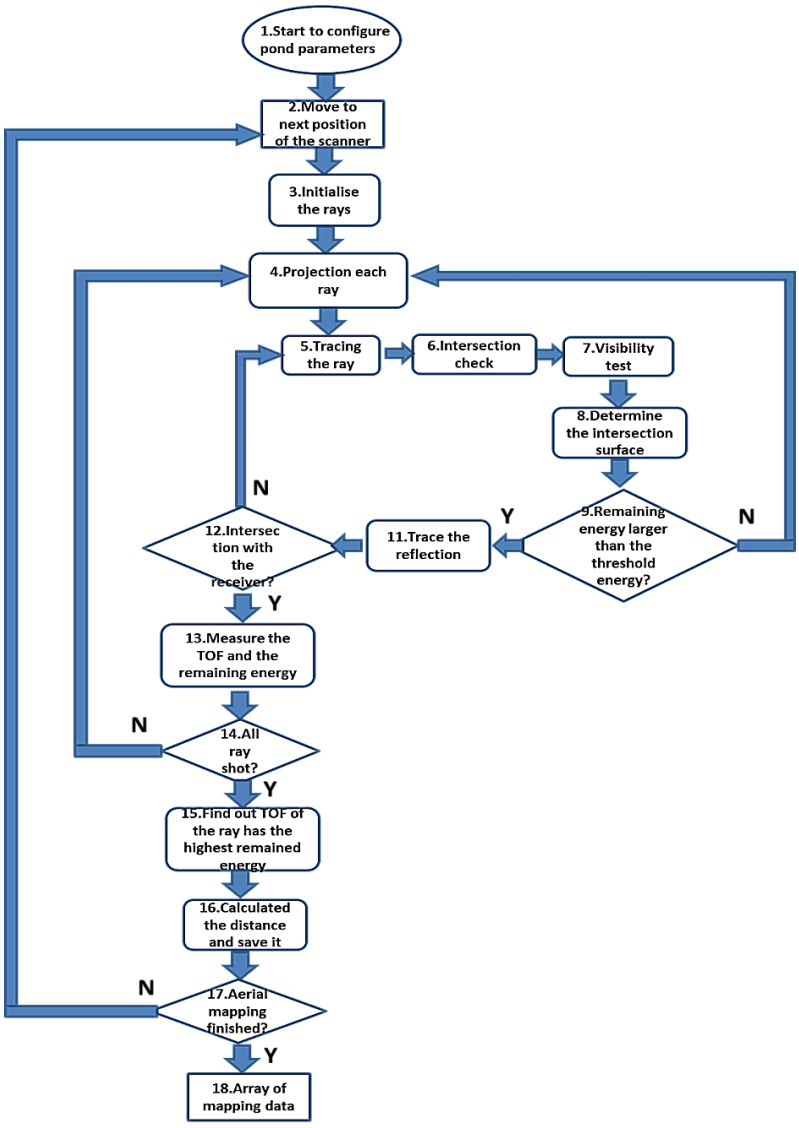

Figure 6. Ray tracing flow chart

A ray-tracing simulator is built on MATLAB. Assume the pond is a 50m*25m swimming pool-sized enclosure with a depth of 10m. Assume the side-scan sampling resolution is 0.5m*0.5m, the obtained depth measurement data and its' corresponding surface plot are shown in Figure 7 and 8 respectively.

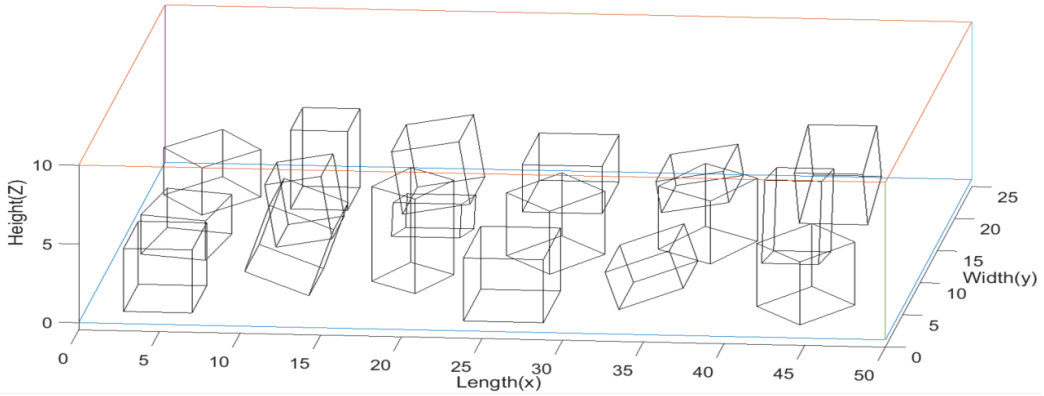

Figure 7. Geometry model of a storage pond

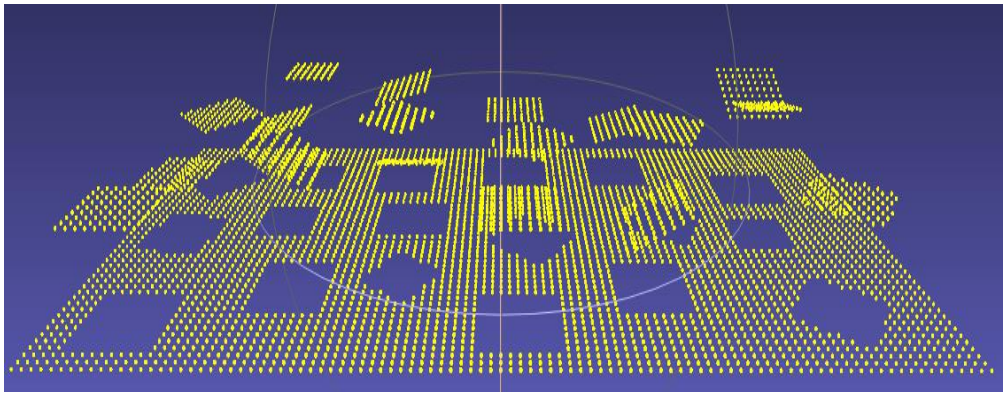

Figure 8. Raw depth measurement points obtained by aerial mapping

3. OCCUPANCY GRID MAP CONSTRUCTING AND PATH PLANNING

The raw data obtained by the aerial mapping is a topological depth measurement data. It is required to be processed with *boundary tracing* (to extract objects' boundary points, estimate objects' occupied region and vertices, and calculate the top plane equation), *Planar depth measurement data regenerating* (to generate depth measurement points that lie on the survey grid based on plane equation estimated by extracted vertices point), *3D structure constructing* (to construct the rest body by interpolating points under the assumption of the canister shape), *and grid method* (to initialize the environment's point cloud matrix by binary variable) in orders to establish the 3D digital grid map for A* path planning algorithm. The flow chart of the processing procedure is shown in Figure 9.

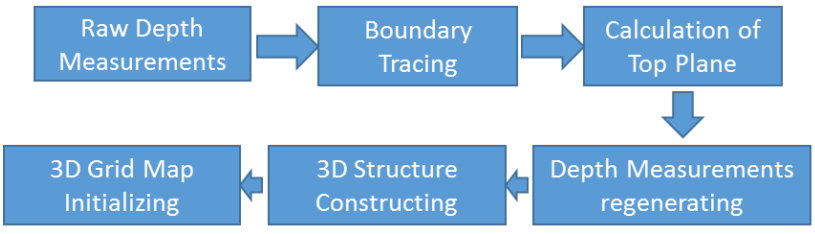

Figure 9. The processing flow chart

3.1. Boundary tracing

Boundary tracing algorithm is a technique that is applied to digital images in order to extract their boundary [21]. A digital image is a group of pixels on a square tessellation each having a certain value. Since point cloud data has similar features so boundary tracing algorithm can be used here. There are 4 common ways to trace the boundary. They are Square Tracing Algorithm, Moore-Neighbor Tracing, Radial Sweep, and Theo Pavlidis' Algorithm [22] separately. They tracking boundary in a different manner. This paper uses the radial sweep algorithm to extract boundary points from point cloud data in order to find vertices points because this algorithm is easy to implement. The boundary tracing includes 2 steps:

- Extract boundary points.
- Removal points of the traced object.

There is a distinguished height difference in the topological point cloud data, which segments ground point sets and objects' point sets. Hence, height checking will proceed in sequence in order to find the first object point which is classified as the origin of the boundary. After the first object point is found, the boundary tracing procedure is invoked. This uses the concept of chain code [23, 24], whose checking encoder moves along the boundary of the region to extract boundary points (in a clockwise or anticlockwise direction). This continues until the checking encoder returns to the starting position.

In order to find the next boundary point, the height of 8 directions points that surrounding the start point will be checked in clockwise order from the south direction until finding the first next boundary point. Diagram of the 8 directions respect to the start point and check direction is shown in Figure 10 (i,j) is used to represent 8 directions of adjacent points relative to the start point.

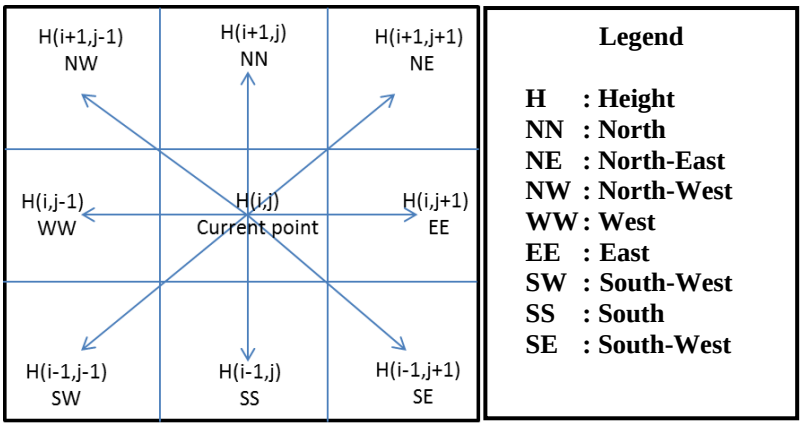

Figure 10. 8 Directions respect to the current point

Once the next boundary point has been found, it is saved and regarded as the new start point and a similar procedure to the above is repeated, with one important difference. The difference is rather than start checking at the south direction it commences from the direction that the previous start point with respect to the current start point. A complete boundary tracing is demonstrated in Figure 11.

As shown in Figure 11, red dots represent the found boundary points and green dots represent the next successive boundary points. The sample data is scanned at black dot from bottom to top searching for the start point. After finding the starting point, start the initial boundary check in the clockwise direction in the south direction of the starting point until the next boundary point is found. After that, regarding the found next boundary point as the new start, and repeat the boundary check at the south-west direction of the new starting point. The boundary checking will terminate when the initial starting point (the first red dot) is found. Figure 11 demonstrates an example of the boundary tracing process on a sample point set. The most important thing in the boundary tracing procedure is the "sense of direction". The right directions are with respect to the previous positioning. Therefore, it's important to keep tracking of the current and previous orientation in order to make the right moves.

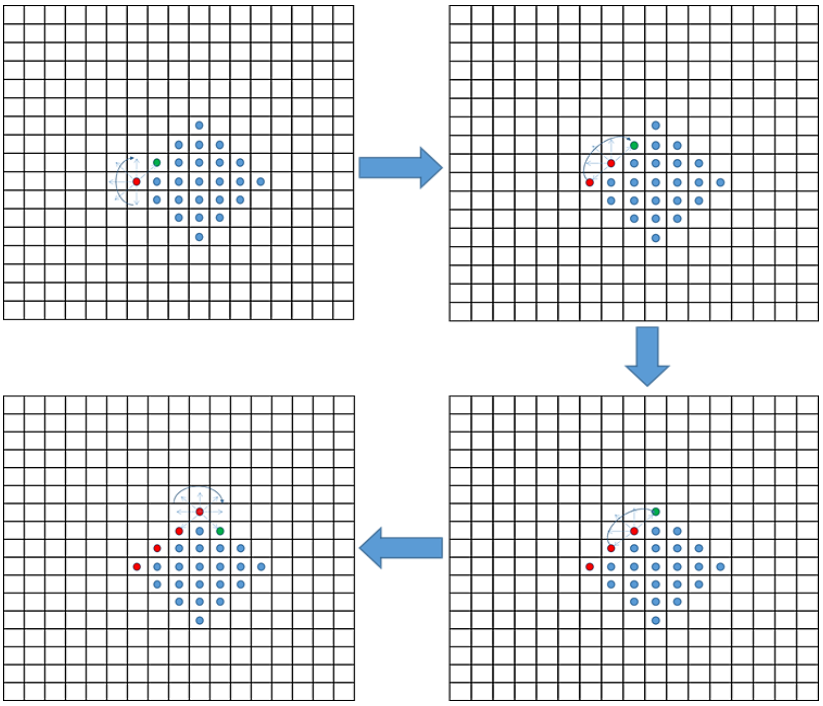

Figure 11. Boundary Tracing

In practice, there will be more than one object in the environment so the obtained point cloud will contain a different set of points (one set of points corresponds to one object). However, the problem is it will repeatedly trace the first object and cannot break out the loop. In order to escape from the loop, it is required to remove each traced object from the point cloud so that it does not prevent tracking of the remaining objects. Once the removal process has been done, restart to find the next object point and trace the successive boundary points point again. See the example in Figure 12.

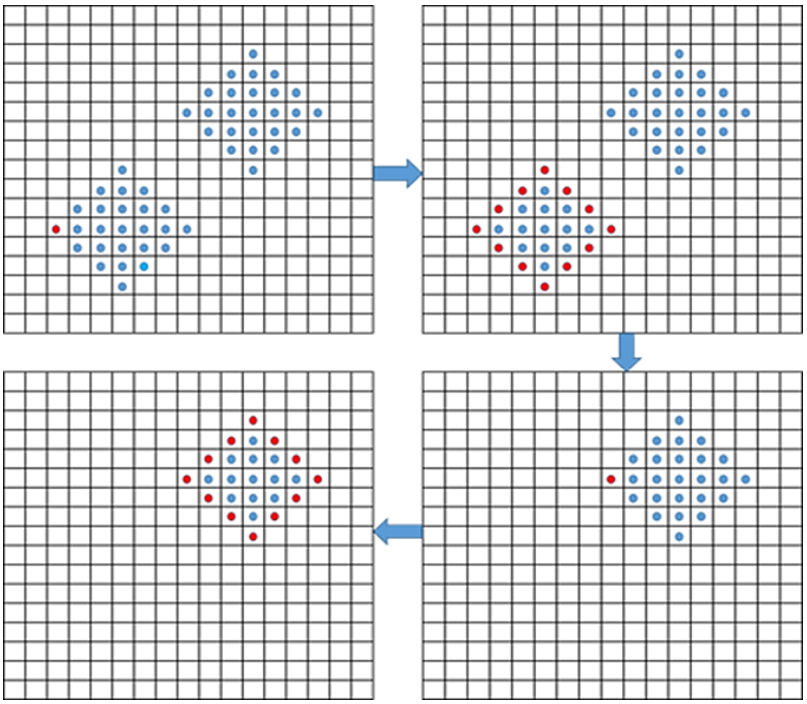

Figure 12. Removal of a traced object

There are 2 sets of points in the point cloud, so there are 2 objects. The red points are boundary points, the blue points are reminder object points and the heights of the black points where scanning starts. Figure 9 demonstrates how to extract boundary points when there are multiple objects in the point cloud. However, it apparently left one problem: When 2 objects overlap or touch each other, the point cloud of their touched edge will be merged together. As a result, this algorithm will treat them as one object. For this particular case, segmentation of point cloud will be deployed to classify this 2 object.

3.2. Environment reconstruction by point cloud

Usually, the raw depth measurement points do not coincide with the user's specified survey grid, so it is necessary to calculate the plane equation to create new measurement points that coincide with the user's specified survey grid.

A plane in 3D space is defined by three points, which do not all lie on the same line or by a point and a normal vector to the plane, which is called the vector equation of a plane. In other words, in order to calculate the vector equation of a plane, it is required to know the coordinate of at least 3 points that located in the plane and not in the same straight line. However, the selection of points needs to be careful, because it cannot be assumed that any 3 arbitrary points taken from the plane are able to represent the plane. The chosen points should be able to best describe the plane parameter. Vertices points are the best option to calculate the plane. Since the boundary points in the point cloud are extracted by boundary tracing, vertices can be found by comparing their x value and y value. Once the coordinate of 3 verticespoints is known, the vector plane equation can be calculated. The general form of the vector equation of a plane is:

$$\vec{n} \cdot (p - p1) = 0 \quad (2)$$

$\vec{n}$ is the normal vector of the plane. The normal vector $\vec{n}$ =(a, b, c) can be calculated by *Cross Product* of any 2 vectors that constituted by the known 3 vertices. P1=(x_1, y_1, z_1) is one of the known vertices points. P=(x, y, z) is any point that located on this plane. This equation can be transformed into the general planar equation, and the height of any point on the plane is given by:

$$[a, b, c] \cdot [x - x_1, y - y_1, z - z_1] = 0$$

$$ax + by + cz - (ax_1 + by_1 + cz_1) = 0$$

$$z = \frac{(ax_1 + by_1 + cz_1) - by - ax}{c} \quad (3)$$

For the case that height measurements at reflection positions that do not lie on the survey grid such as tilted objects, this vector plane equation (3) can be used to calculate the height of the points whose position lie on the survey grid and the object's top plane (x and y value is the coordinate of the corresponding survey grid). Thus, acquired a new point cloud that denotes the corresponding object' top plane by points that lie on the survey grid. Since there are numbers of objects, the above processes will be undertaken for each object.

After regenerating the new measurement points of the object' upper plane, the next step is to further process it to create a completed 3D model of the environment. This procedure is called environment reconstruction. As objects are assumed to be prismatic, so their side surfaces are perpendicular to its upper surface. This a crucial assumption for reconstructing the object because the key idea of reconstructing the environment by Point Cloud is to add new points to the modified measurement points in the perpendicular direction of the upper plane to create the remaining part between the top plane depth measurement data of the obstacle and the ground measurement data. Added points must satisfy the condition that lies on the user's specified survey grid. Thus, it generates the 3D point cloud model of obstacles. This technique is known as linear height interpolation [25]. The schematic diagram of the perpendicular height interpolation is shown in Figure 13.

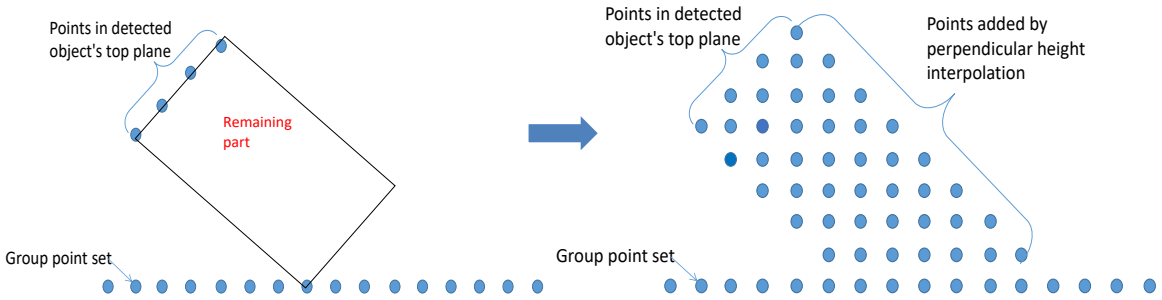

Figure 13. A schematic of the linear perpendicular height interpolation

An example of using linear perpendicular height interpolation to generate a point cloud of the 3D model of detected objects is shown below. It is explained with the aid of Figure 14.

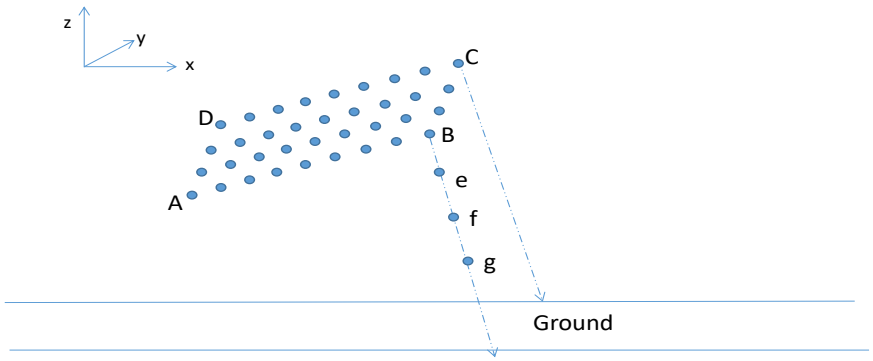

Figure 14. A schematic of the linear perpendicular height interpolation

Assume the resolution of the survey grid is α*α, and the position of measurement result lies on the survey grid. Point A, B, C, D are 4 vertices points of the measurement point cloud. Assume the coordinate of A is (x_a, y_a, z_a), B is (x_b, y_b, z_b), C is (x_c, y_c, z_c), D is (x_d, y_d, z_d) respectively. It needs to calculate the x, y coordinate of added points e, f, g and make sure they are lying on the survey grid. Point e, f, g exist on the vector that perpendicular to the plane and pass through B, so it needs to calculate the equation of the vector. The equation of the vector and the coordinate of the point in the vector is given by:

$$\overrightarrow{AB} = [x_b - x_a, y_b - y_a, z_b - z_a]$$

$$\overrightarrow{AD} = [x_d - x_a, y_d - y_a, z_d - z_a]$$

$$\vec{n} = \overrightarrow{AB} \times \overrightarrow{AD} = [x_n, y_n, z_n] \quad (4)$$

$$\vec{p} = \vec{B} + t \cdot \vec{n} = [x_b, y_b, z_b] + t \cdot [x_n, y_n, z_n] \quad (5)$$

$$[x, y, z] = [x_b + t \cdot x_n, y_b + t \cdot y_n, z_b + t \cdot z_n] \quad (6)$$

$$x = x_b + t \cdot x_n \quad (7)$$

$$y = y_b + t \cdot y_n \quad (8)$$

$$z = z_b + t \cdot z_n \quad (9)$$

P is any point that crossed by this vector. Assume z coordinate of e z_e is known (the interpolated height is from α to the height (z-axis value) of B, the interval is α which equals to the z-axis interval of the A* searching grid), so y_e and x_e are given by:

$$e = [x, y, z_e] \quad (10)$$

$$z_e = z_b + t \cdot z_n \quad (11)$$

$$t = \frac{z_e - z_b}{z_n} \quad (12)$$

$$y_e = y_b + \frac{z_e - z_b}{z_n} \cdot y_n \quad (13)$$

$$x_e = x_b + \frac{z_e - z_b}{z_n} \cdot x_n \quad (14)$$

The remaining question is to determine which grid cell is that point e locates in. In order to do so, it is necessary to find its nearby vertex on the survey grids to determine which grid cell is that point **e** locates. If the position (x_e, y_e) of e is not on the survey grid but very close, it should be assigned to the nearest survey grid position. After processing the new topological point cloud by procedures given above, a 3D point cloud model of detected objects will be obtained. Each point is assigned to the corresponding grid in the A* survey grid.

3.3. Grid method to establish 3d occupancy grid map

Since the 3D model of the environment is represented by discrete points in the user's queried 3D grid (A* searching grid), a 3D digital map is able to be established by grid method [26]. The basic idea of the occupancy grid method is to create an environment map as an evenly spaced field of binary numbers: obstacle regions and free regions are encoded by 0 and 1 respectively. First of all, divide the whole environment into a finite number of equal spaced searching grids without considering obstacles as shown in Figure 15 and 16.

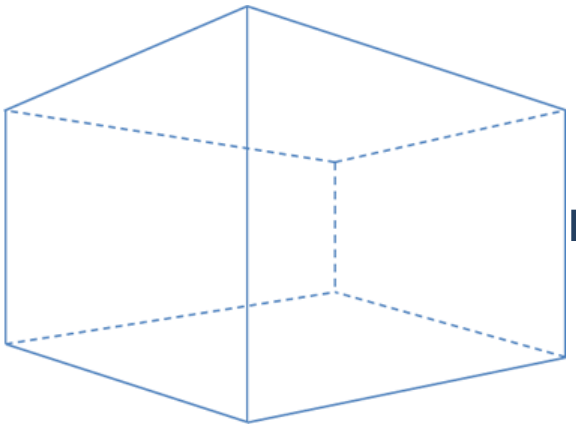

Figure 15. The whole environment

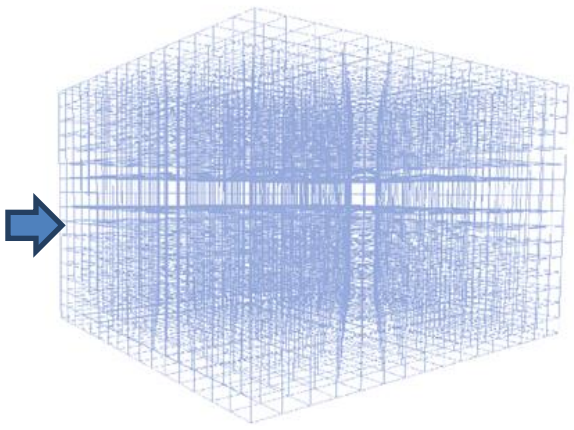

Figure 16. The meshed environment

Then, model obstacles by a finite number of discrete equal spaced cells. After that, align each point to its corresponding survey grid and convert it to the occupancy grid map. In order to convert the point cloud model to a 3D digital grid map, the first step is to define cells occupied by objects as inaccessible cells and cells not occupied by points as accessible cells. The accessible cells are outside obstacles; while the inaccessible cells are inside obstacles. Next, initialize the 3D matrix that will hold the digital map by encoding each inaccessible cell with '1' and accessible cell with '0'. As a result, the binary matrix of the 3D digital grid map of the environment is obtained. It is ready to be used in path planning algorithms, e.g. A* algorithm. By this method, the raw depth measurement data shown in Figure 5 can be used to construct 3D point cloud model of the pond, and the point cloud model can be converted to a 3D occupancy grid map, as shown in Figure 17 and 18. Figure 17 is the 3D point cloud structure of the environment after processed by

3D structure reconstruction. Each point will be assigned to the meshed environment to establish an occupancy grid map. Figure 18 shows a well-formatted 3D digital grid map, the red dots are an unoccupied grid, which encoded as '0', and the yellow dots are occupied grid, which encoded as '1'.

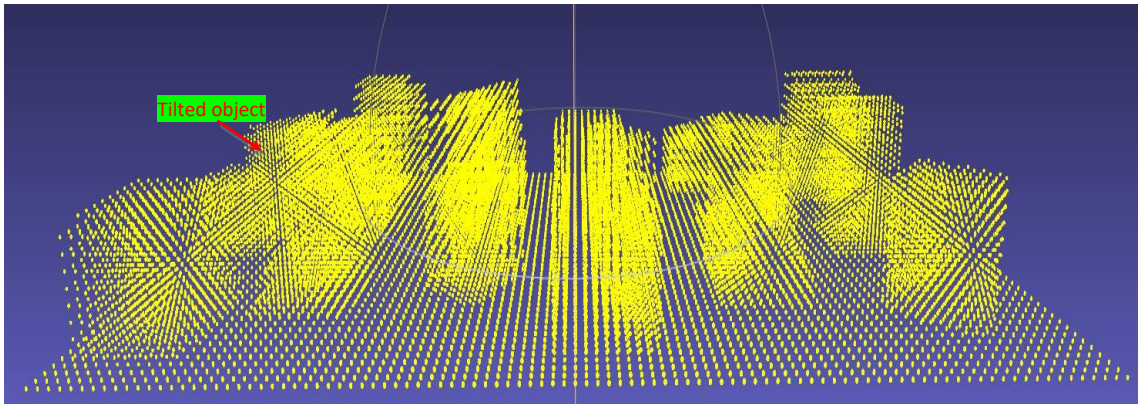

Figure 17. Height interpolated 3D point cloud

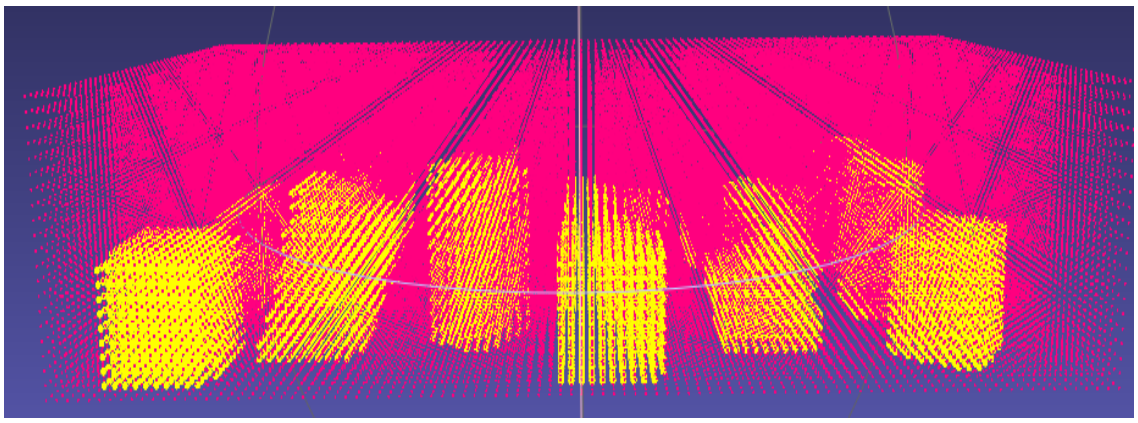

Figure 18. 3D grid map

3.4. A* Path planning on the obtained map

A* algorithm [27] are typical search algorithm which is widely used in pathfinding and graph traversal. The main reason for choosing it is this algorithm is it can search a short and collision-free path on discrete sample nodes based workspace. The basic principle of this algorithm is continuously searching the appropriate next node to approach the destination by considering the heuristic information and the condition of the current node. A node is a grid point which represents a small piece of the area of the workspace, and the workspace consists of a finite number of nodes. The established occupancy grid map can be easily treated as the A* algorithm's workspace. The mathematical formula of this algorithm is expressed in function (15) below:

$$F(n) = g(n) + h(n) \quad (15)$$

Where n is the last node on the path; *h(n)* is a heuristic function which estimates the cost of the shortest path from the current node to the goal node; g(n) is a movement function which estimates the cost from the start to the current node; *F(n)* is the cost function which estimates the total cost of moving from the start node to the target node. Usually, the Euclidean distance [28] is used to estimate the cost between the current node and the goal node, so the heuristic function *h(n)* can be expressed:

$$h(n) = \sqrt{\left(x_{current} - x_{goal}\right)^2 + \left(y_{current} - y_{goal}\right)^2 + \left(z_{current} - z_{gosl}\right)^2} \quad (16)$$

The collision-free path obtained by A* algorithm on the occupancy grid map in Figure 18 is given by Figure 19 and 20, the μAUV is assumed as a moving particle with 6 degrees of freedom without considering its size:

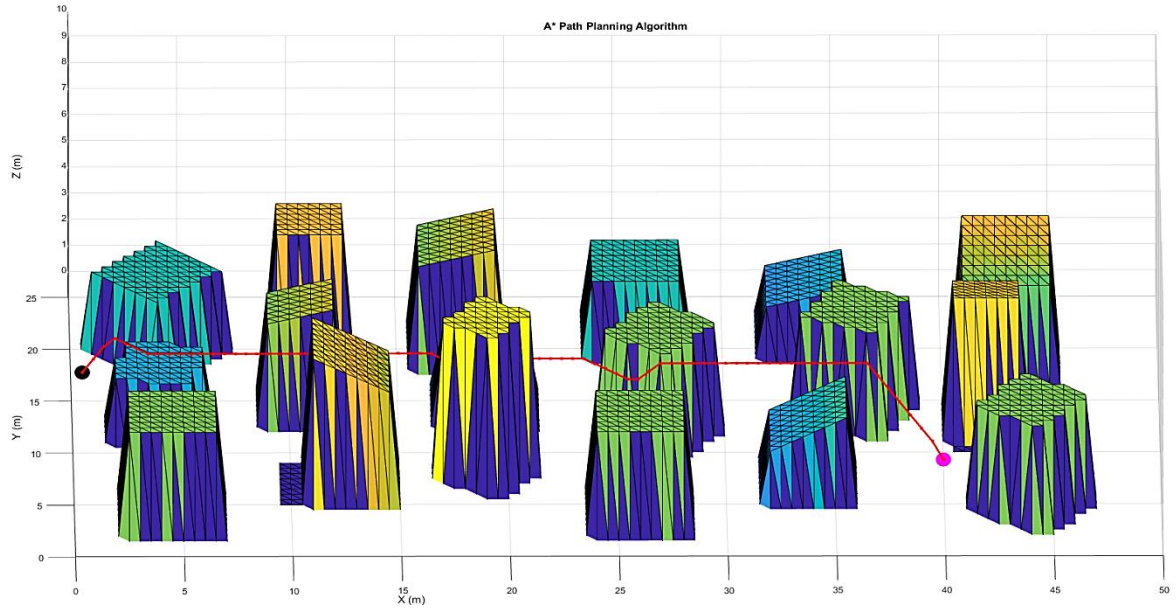

Figure 19: 3D view of planned A* path

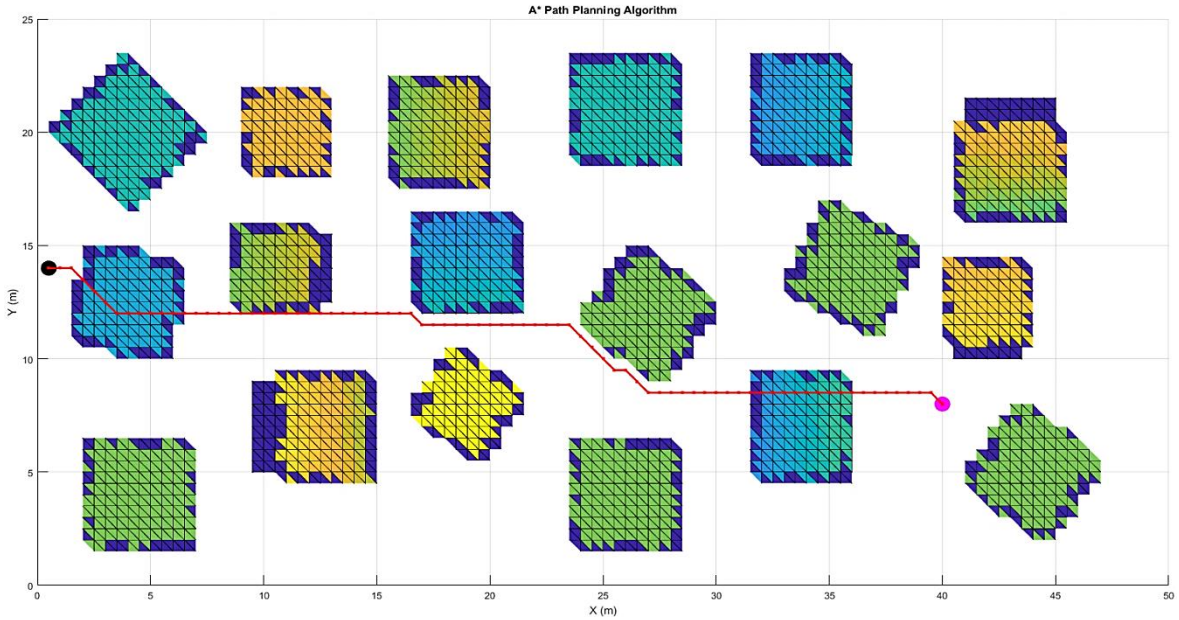

Figure 20: Plan view of planned A* path

Figure 7, 9, 17, 18, and 19 successfully demonstrated a simulation that μAUV performing aerial mapping to the depth measurements of the pond environment, processing the obtained depth measurements data to create a point cloud model of the pond, constructing an occupancy grid map based on the point cloud model, and planning a collision-free path by A* algorithm on the obtained occupancy grid map. It has been

observed from the above figures that the approach proposed in this paper provides a reliable collision-free path for a µAUV work in a cluttered storage pond. The advantage of this method compared to the method given in [29] is that it can be used for a sensor limited low-cost µAUV.

4. DISCUSSION OF ERRORS IN MEASUREMENTS

Measurement error is an important factor that needs to be considered in pond data acquisition and path planning. In practice, the depth measured by the acoustic sensor will not be the same as the true depth. The accuracy might vary from a few centimeters up to 10 centimeters depending on the quality of the sensor. Measurement errors in height will cause the obtained 3D point cloud model of the tilted object to be shifted in the horizontal plane relative to the real position. As a result, the measured object will be either smaller or bigger than the true object, which may cause collision problem when used for planning a collision-free path in practice. Therefore, it is necessary to create a larger bounding box that can completely cover the real object based on the measurement data, as shown in Figure 21.

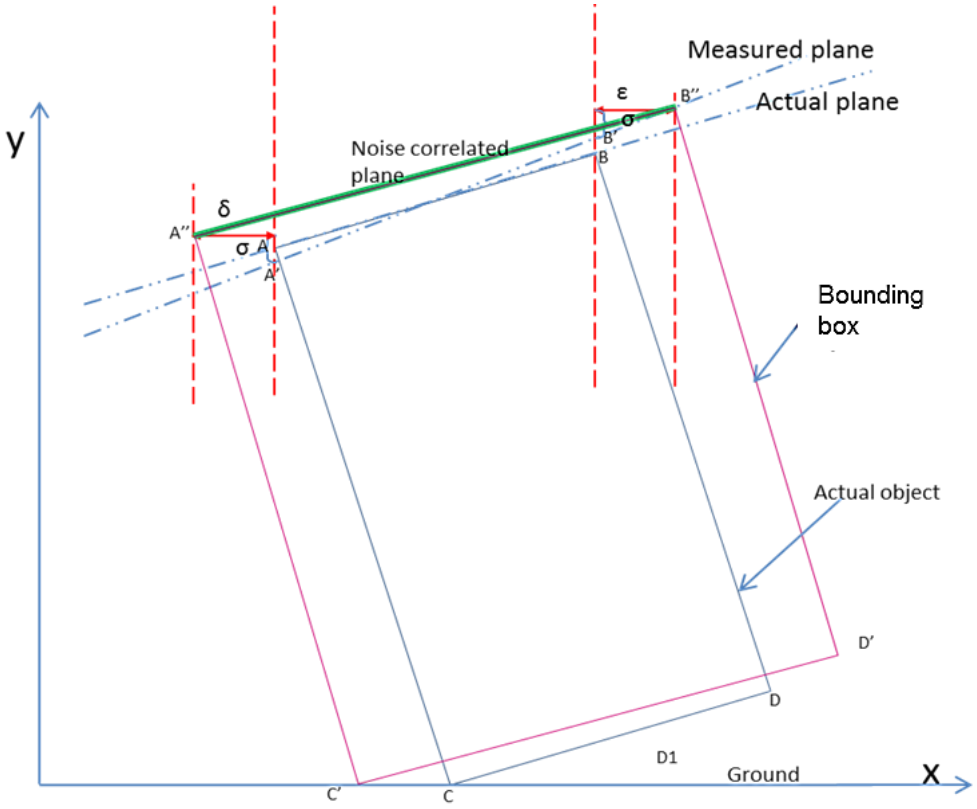

Figure 21. Bounding box

Assuming the acoustic sensor noise has a Gaussian distribution and its standard deviation is σ. measured vertex $A' = (x_{a'}, y_{a'}), b' = (x_{b'}, y_{b'})$. δ is the maximum separation caused by Gaussian distribution in the left-hand side of the object; ε is the maximum separation caused by Gaussian distribution in the right-hand side of the object. So the boundary vertex of the bounding box's upper plane is given by:

Left boundary vertex $A" = (x_{a'} - \delta, y_{a'} + 2\sigma)$ (17)

Right boundary vertex $B" = (x_{b'} + \varepsilon, y_{b'} + 2\sigma)$ (18)

Based on the approach introduced in section 2 of this paper, the point cloud model of and occupancy grid map of the bounding box can be constructed. Figure 22 shows paths calculated by the A* algorithm in the workspace of the measurement object and the workspace of the bounding box.

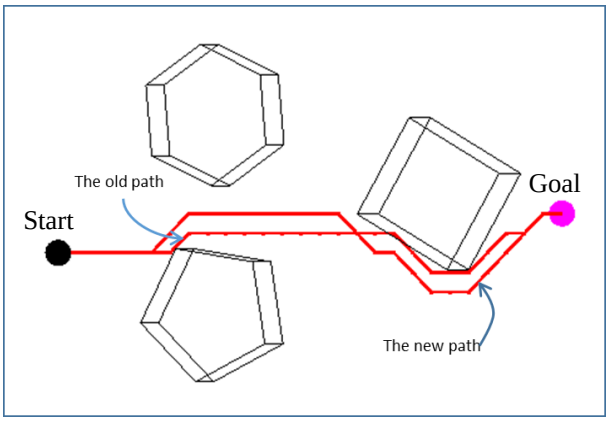

Figure 22. Comparison of planned path

The old path is the path calculated for the measured object, and the new path is the path calculated for the bounding box. The advantage of the bounding box method prevents the path from being too close to obstacles, which addressed the collision problem when used for planning a collision-free path in practice.

5. CONCLUSION

The main contribution of this paper is to investigate the navigation strategy and path planning of micro autonomous underwater vehicles (µAUVs) which is used to replace human being to work, monitor, and take measurements in an enclosed cluttered underwater environment such as the nuclear storage pond. This kind of missions requires the µAUV to map the working environment, construct a geographic map for visualising and grid occupancy map for path planning, and plan a feasible collision-free path on the grid occupancy map. The scope of this research includes how to obtain a point cloud map of the proposed enclosed cluttered underwater environment, study the feasibility of available navigation and path planning algorithm for an enclosed cluttered underwater environment, how to implement path planning algorithms to the point cloud map.

The research conducted throughout this paper leads to an algorithm that capable of reconstructing the aerial mapping point cloud map to an occupancy grid map and using for path planning. However, there are a lot of unexplored areas need to be addressed in the future, that can extend what has been presented here. First of all, the online path planning, since the aerial mapping is unable to gather the information below the shadow area or overlap area, when the µAUV travelling on the predefined trajectory, the simultaneous localization and mapping (SLAM) technology [3] can be used on the µAUV to minor correct the planned trajectory in real-time. The second topic of future work is taking the speed of µAUV into considering. The frameworks proposed in this paper assume the µAUV move in a constant speed and maximum turning rate. In this way, the AUV path can be parameterized with a straight line segment and circular arc with any radius. However, in practice, the speed of the µAUV should be considered in case of sharp turning. The work on motion speed control should be investigated in order to find a feasible and safe path with speed varying.

REFERENCES

[1] S. L. Sharma, A. Qavi, and K. Kumari, "Oil Pipelines/Water Pipeline Crawling Robot for Leakage Detection/Cleaning of Pipes," *Global Journal of Researches in Engineering*, vol. 14, no. (1), 2014.
[2] S. Nawas, M. Hussain, S. Watson, N. Trigoni, and P. N. Green, "An Underwater Robotic Network for Monitoring Nuclear Waste Storage Pools," *International Conference on Sensor Systems and Software*, vol. 24, pp. 236-255, 2009.
[3] R. B. Wynn *et al*., "Autonomous Underwater Vehicles (AUVs): Their Past, Present and Future Contributions to the Advancement of Marine Geoscience," *Mar. Geol*., vol. 352, pp. 451–468, 2014.
[4] T. Paterson, "Malaysia Airlines: World's Only Three Abyss Submarines Readied for Plane Search,". *Telegraph.co.uk*. 23 March 2014.
[5] "Storage of Water Reactor Spend Fuel in Water Pools," *International Atomic Energy Agency*, 1982.
[6] World Nuclear Association, "Radioactive Waste Management," *World Nuclear Association*, 2015 updated.
[7] Picture from "News story Game-changing Progress in Sellafield pond," *Sellafield Ltd and Nuclear Decommissioning Authority*.
[8] S. Watson, "Mobile Platforms for Underwater Sensor Networks," *Ph.D. dissertation, Electrical & Electronic Engineering School, The University of Manchester*, 2012.

[9] A. Griffiths, A. Dikarev, P. R. Green, B. Lennox, X. Poteau, and S. Watson, "AVEXIS—Aqua Vehicle Explorer for In-Situ Sensing," *IEEE Robotics and Automation Letters*, vol. 1, no. (1), Jan. 2016.
[10] L. Paull, S. Saeedi, M, Seto and H. Li, "AUV Navigation and Localization: A Review," *IEEE J. Ocean*, vol. 39, no. (1), 2014.
[11] M. Liu, "Robotic Online Path Planning on Point Cloud," *IEEE Transaction on Cybernetics*, vol. 46, no. (5), 2016.
[12] B. A. Merhy and P. Payeur, "Application of Segmented 2D Probabilistic Occupancy Maps for Robot Sensing and Navigation," *IEEE Transaction on Instrumentation and Measurement*, vol. 57, no. (12), December 2008.
[13] H. Temeltas and D. Kayak, "SLAM for Robot Navigation," *IEEE Aerospace and Electronic Systems Magazine*, vol. 23, no. (12), December 2008.
[14] L-3 Communications SeaBeam Instruments, "Multibeam Sonar - Theory of Operation," *L-3 Communications SeaBeam Instruments*, 2000.
[15] H. Cho, B. Kim, and S. C. Yu, "AUV-based Underwater 3-D Point Cloud Generation using Acoustic Lens-Based Multibeam Sonar," *IEEE J. Ocean. Eng*., vol. 43, no. (4), pp. 856–872, 2018.
[16] C. De Moustier and H. Matsumoto, "Seafloor Acoustic Remote Sensing with Multibeam Echo-Sounders and Bathymetric Sidescan Sonar Systems," *Mar. Geophys. Res*., vol. 15, no. (1), pp. 27–42, 1993.
[17] J. Melo and A. Matos, "Survey on Advances on Terrain Based Navigation for Autonomous Underwater Vehicles," *Ocean Engineering*, pp 250-264, 2017.
[18] J. Kirschner, "Surface Acoustic Wave Sensors (SAWS): Design for Application," *Micro-Electronical Systems*. Dec. 2010.
[19] N. Röber, U. Kaminski, and M. Masuch, "Ray Acoustics Using Computer Graphics Technology," *Conference on Digital Audio Effects (DAFx-07)*, Sep 10-15, 2007.
[20] D. O. Elorza, "Room Acoustics Modeling using the Raytracing Method: Implementation and Evaluation". *University of Turku Department of Physics*, 2005.
[21] Y.-H. Tseng and H.-C. Hung, "Extraction of Building Boundary Lines from Airborne LIDAR Point Clouds," The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences *Congress*, vol. XLI-B3, July 2016.
[22] P. R. Reddy, V. Amarnadh, and M. Bhaskar, "Evaluation of Stopping Criterion in Contour Tracing Algorithms,". *International Journal of Computer Science and Information Technologies*, vol. 3, pp. 3888–3894, 2012.
[23] F. Baji and M. Mocanu, "Chain Code Approach for Shape Based Image Retrieval," *Indian J. Sci. Technol*., vol. 11, no. (3), pp. 1–17, 2018.
[24] W. Shahab, H. Al-Otum, and F. Al-Ghoul, "A Modified 2D Chain Code Algorithm for Object Segmentation and Contour tracing,*" Int. Arab J. Inf. Technol*., vol. 6, no. (3), pp. 250–257, 2009.
[25] D. Kidner, M. Dorey, and D. Smith, "What's the Point? Interpolation and Extrapolation with a Regular Grid DEM," *Division of Mathematics & Computing, School of Computing, University of Glamorgan*.
[26] A. Bücken and S. Thrun, "Integrating Grid-Based and Topological Maps for Mobile Robot Navigation," *Artif. Intell*, pp. 944–951, August 1996.
[27] P.E. Hart, N. J. Nilsson, B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics*, vol: 4, Issue: 2, July 1968.
[28] D. J. Peuquet, "An algorithm for Calculating Minimum Euclidean Distance between Two Geographic Features," *Comput. Geosci*., vol. 18, no. (8), pp. 989–1001, 1992.
[29] J. D. Hernández *et al*., "Autonomous Underwater Navigation and Optical Mapping in Unknown Natural Environments," *Sensors (Switzerland)*, vol. 16, no. (8), 2016.