Hittite Journal of Science and Engineering, 2020, 7 (2) 125-134
ISSN NUMBER: 2148-4171
DOI: 10.17350/HJSE19030000181

# A Novel Navigation Algorithm for Mapping Indoor Environments with a Quadrotor

**Omer Oral[1] Ali Emre Turgut[1] Kutluk Bilge Arikan[2]**
[1]Middle East Technical University, Department of Mechanical Engineering, Ankara, Turkey
[2]TED University, Department of Mechanical Engineering, Ankara, Turkey

**ABSTRACT**

In the last decade, unmanned aerial vehicle gained popularity and started to be used in different tasks most of which are performed in outdoor environments. Still, there is a great potential to use quadrotors in indoor tasks such as urban relief and disaster operations. In this paper, we developed a framework and a novel target-based navigation algorithm for mapping of an unknown 2D environment with a quadrotor using an ultra-wideband system. The target-based navigation algorithm aims to explore map of the environment by moving the border between the discovered and undiscovered areas. It uses A* search algorithm for path planning if there is an obstacle present in the environment. The target-based navigation algorithm is implemented on Gazebo simulator and its performance is compared with the well-known wall following algorithm and exploration algorithm in terms of task completion time and distance travelled. The target-based navigation algorithm outperforms the other two algorithms especially in environments with obstacles.

***Article History:***
*Received: 2020/03/04*
*Accepted: 2020/05/22*
*Online: 2020/06/26*

***Correspondence to**: Omer Oral,*
*Middle East Technical University,*
*Department of Mechanical Engineering,*
*06800, Ankara, TURKEY*
*E-mail: omroral@gmail.com*

**Keywords:**
LIDAR; SLAM; Mapping; UWB; Trilateration technique; Quadrotor; UAV; Indoor; Obstacle avoidance.

## INTRODUCTION

Recently, quadrotors gained popularity due to their high maneuverability, cost and vertical take-off/landing capabilities. Nevertheless, they also have disadvantages such as they have a limited flight time and small payload capabilities. In addition to these, a major part of the energy of a quadrotor is spent against gravity for hovering. Still, they are one of the most adopted air vehicles for commercial and research purposes. Most of the time UAV's (Unmanned Aerial Vehicles) are used in outdoor applications such as surveillance, search/rescue and patrolling. Recently, UAV's started to find uses in indoor environments. Material handling in manufacturing and inspection in harsh environments are to name a few[1] of these applications. One of the most promising applications is to utilize them in urban relief and disaster operations where a UAV moves autonomously avoiding obstacles in GPS-denied buildings to help human operators for rescue operations.

In order to navigate in an indoor environment, a map of the environment is needed. However, in most of the cases either the map is unknown or some partial information about the environment is available. Indoor mapping process can be performed by using several methods. SLAM technique is usually employed for mapping. SLAM (Simultaneous Localization and Mapping) is the process of simultaneous map extraction and robot localization. The difficulty of this process is inherent in its definition. A map is required for correct localization, while an accurate localization is required for mapping [2]. In indoor environments, ground-based robots were commonly employed for map extraction [3]. In studies with ground robots, while LIDAR (Laser Imaging Detection and Ranging) and IMU (Inertial Measurement Unit) were mostly used, Omara et al. [4] used a Kinect sensor for exploration and mapping of the environments using SLAM methods. Different sensors were utilized when UAV's were used for mapping. John son [5] used IMU, optic flow sensors and a camera to navigate autonomously in indoor environments. Due to unreliable attitude estimations with inertial sensors, Hough transform was adopted for attitude estimation of the quadrotor. Ahrens et al. [6] also used a quadrotor equipped with IMU and a camera to navigate in indoor environments with obstacles. They used "good features to track" detector for feature tracking and use these features for navigation and mapping. Roberts et al. [7] developed a quadrotor capable of navigating in indoor environments autonomously using and IMU, an ultra-

sonic sensor and four infrared sensors. The system managed to fly indoors with success. Mohamed et al. [8] proposed an indoor navigation system to estimate the position and orientation of the UAV. In this system, three laser diodes were integrated to the body of the UAV and they generate three dots on the ground. These dots were captured by the camera and their coordinates were determined using Scale Invariant Feature Transform algorithm, hence the position and orientation of the UAV were determined. Achtelik et al. [9] developed a method for autonomous navigation and exploration in indoor environments. They used an odometry algorithm with Extended Kalman Filter to estimate the position of the quadrotor and they employed a SLAM algorithm and a mission planner for building the map of the environment. Parallel to Achtelika's work, Grzonka et al. [10] performed mapping and exploration tasks using a UAV, an IMU and a LIDAR. A mirror was used to reflect laser beams to the ground and height of the air vehicle was determined accordingly. Wang et al. [11] presented a method for navigation and control for a UAV operating in an indoor environment. The quadrotor was equipped with an IMU, a downlooking camera, a barometer for height measurement, and a LIDAR to get distance information. The robot was able to estimate its position and velocity using Kalman filter, and fly in the room without colliding any walls. Wall following was used for navigation.

In the studies discussed so far, IMU and LIDAR were mostly used for localization. However, in some recent studies, wireless localization systems started to be employed. One of the widely used wireless localization technology for indoor environments is the UWB(Ultra-wideband) systems. UWB is an old radio technology used for short-range wireless communication that utilizes frequencies from 3.1 to 10.5 GHz and supports a bandwidth of 500 MHz. Due to its high bandwidth, very high data rates can be achieved with UWB systems. The only drawback of UWB is its limited communication range. Recently, due to the advances in technology, UWB is started to be used for real-time localization and tracking of objects in indoor environments by making time of flight measurements. In UWB-based localization, trilateration technique is used where a UWB tag is placed on the robot and at least three UWB anchors are placed at known positions in the environment [12]. In this way, the robot calculates its position with respect to the anchors [13,14]. UWB-based localization systems are capable of detecting robots with a 10 cm accuracy that are moving at a speed of 20 km/h. Barral et al. [15] developed a system for tracking forklifts using UWB technology in an indoor environment. They tested various real world scenarios on Gazebo simulator by defining different modes of operation.

The aim of this paper is to develop a framework and propose a method to obtain the 2D map of an unknown indoor environment with a quadrotor using an UWB system. We improved the state-of-the-art in two different aspects: First, to the best of our knowledge, this is the first implementation of a UWB system with a UAV for mapping an indoor environment. Second, we proposed a novel indoor navigation algorithm for mapping with a UAV and compared its performance with the two well-known algorithms.

## MATERIAL AND METHODS

First, we present the localization and mapping techniques. Following that the three different navigation algorithms including our novel navigation algorithm are discussed. Lastly, the experimental setup is introduced.

### Localization and Mapping

The most important and challenging part of this study is to obtain the location of the UAV and the map of the environment simultaneously. The aim of the SLAM process is to update the position of the robot by exploring the environment. SLAM operation consists of several stages. In general, odometry information calculated from the IMU is used to estimate the position of the UAV. However, this information is inaccurate due to noise and drift in IMU's. Therefore, we used a UWB localization system to estimate the position of the UAV. A LIDAR is also used both to reduce the error in position estimation and to detect the walls and obstacles in the environment. During mapping, the walls and obstacles present in the environment are extracted and observed. When the robot moves, observations are redone at the new position of the robot. Then, the robot tries to associate the new observations with the previous ones. Reobserved landmarks are used for updating the position estimation while the new landmarks are stored and used in the next steps. The UWB localization system is used together with the SLAM process and it is responsible for updating the position of the robot and the map of the environment.

### Navigation Algorithms

In this section, the three navigation algorithms including our novel navigation algorithm are introduced.

### Wall Following Algorithm

Wall following algorithm is a simple but highly effective method used for exploring and mapping of indoor environments [16]. Details of the wall following algorithm is discussed in a related study [17]. There are two different versions of the wall following algorithm in which the robot moves to the right or left of the wall. In this study, we employed the right version of the algorithm as detailed in Algorithm 1.

**Algorithm 1.** Pseudocode for Wall Following Algorithm [17].

```
while True do
    if Wall is not sensed ahead of the UAV then
        Adjust the turning velocity;
        Adjust the forward velocity;
        Adjust the altitude;
    else
        Turn 90 degrees right
    end
end
```

## Exploration Algorithm

The purpose of this algorithm is to explore a closed area while avoiding collisions and covering the minimum possible distance. The total velocity of the UAV is obtained by the vectorial addition of the velocity contributions from the opening detection and obstacle avoidance behaviours [17].

The opening detection behaviours is used to identify the openings in the environment and to select the next destination point accordingly. The opening points are obtained by comparing the proportions of distances obtained from two consecutive angles over the entire scanning area. The distance of the opening from the station where the robot stops is found using the following formula:

$$r_{corner} = max \frac{r_i}{r_i + 1}, i = 1,2,3...,270 \quad (1)$$

where $r_i$ and $r_{i+1}$ are two consecutive laser scans while $r_{corner}$ is the detected distance from the station.

Since the UAV's orientation, estimated position, UWB data, distance to the corner point and angle information are known, the position of the corner point can be obtained using Eq. 2 and Eq. 3.

$$x_{corner} = x_{UAV} + r_{corner} cos\ (\psi + \theta) \quad (2)$$

$$y_{corner} = y_{UAV} + r_{corner} cos(\psi + \theta) \quad (3)$$

where, $x_{corner}$ is the x coordinate of the destination point and $y_{corner}$ is the y coordinate of the destination point, $\psi$ and $\theta$ refer to the heading angle and the angle between corner and quadrotor, respectively. Then, the angle between the target and the UAV is selected to be 10° away form the corner compared to the the angle between the corner point and the UAV. While the UAV's orientation is always towards the target, its forward speed is adjusted according to its distance from the target as calculated in the Eq. 4. Here, $v_{forward}$ is the forward velocity. $K_0$ and $K_p$ represents the predefined velocity gain coefficients and $l$ is the distance to destination point. The reason why using the cube root of the distance is to obtain smoother velocity changes as quadrotor gets closer to the destination.

$$v_{forward} = K_0 + K_p \sqrt[3]{l} \quad (4)$$

The speed contribution required to move around the obstacles safely and to move around the walls without collisions is calculated using the information obtained from the laser range finder at each angle. Using laser scans at each angle, velocity vectors are generated such that the magnitude is inversely proportional to and opposite to the distance as:

$$v_i = \frac{K}{r_i} \hat{r_i} \quad (5)$$

where, $K$ is the predefined obstacle avoidance gain and $v_i$ is the obstacle avoidance velocity. By summing the velocity vectors from the opening detection algorithm and obstacle avoidance algorithm, the velocity of the UAV is calculated.

## Target-Based Navigation Algorithm

Target-based navigation algorithm is based on "Frontier-based Probabilistic Approach" algorithm [18]. This algorithm aims to explore map of the environment by moving the border between the discovered and undiscovered areas.

Basically, the target-based navigation algorithm creates an occupancy matrix by using data coming from the sensors. In this matrix, unknown areas take "-1" value and known regions take a value between "0" and "100" according to the occupancy rate. The Algorithm 2 first finds potential points to move using the Algorithm 3 and adds these points to a list. Then, it selects the nearest destination point and creates a path from the UAV to the target point. If the target point is reachable, UAV flies directly to that point as in the Algorithm 4. If there is an obstacle between the current position of the UAV and the next destination point, it uses A* algorithm to find the shortest obstacle-free path to the destination point. The algorithm monitors if the UAV reaches the destination point, and gives the next destination point until all the region is explored.

**Algorithm 2.** Pseudocode for Navigation Controller Algorithm Main Function.

```
prevNavStatus = DestNotReached;
NavStatus = DestNotReached;
while True do
    if NavStatus == DestReached && prevNavStatus == DestNotReached then
        Add current destination to AlreadyVisitedPoints;
        FindOpenSpaces(); //Reexamine the map;
        if PointsToBeVisitedList.empty() then
            Print a Message that all of the map is visited;
            return 0;
        else
            SelectNextDestination();
        end
    end
end
```

**Algorithm 3.** Pseudocode for Navigation Controller Algorithm FindOpenSpaces Function.

```
For each Grid Point on currentMap;
CheckOccupancy();
CheckClearance();
CheckAlreadyVisitiedList();
CheckToBeVisitiedList();
if Point meets criteria listed above then
    Add the point to PointsToBeVisited list;
end
```

**Algorithm 4.** Pseudocode for Navigation Controller Algorithm SelectNextDestination.

```
NearestPoint = FindTheNearestPointToBeVisited(); // Gets the nearest point in the
  PointsToBeVisitedList;
if isFlyable(CurrentPoint,NearestPoint) then
    NextDestination = NearestPoint; // If the nearest point is directly flyable, then it is the
      next destination;
    publish(NextDestination); // Send it to the FlightController;
else
    SelectNextDestinationByAStar(); // If the nearest point is not directly flyable, let A*
      algorithm decide the next destination point;
end
```

A* search algorithm [19] calculates the cost of each adjacent point by using an heuristic evaluation function as in Eq. 6. After all the calculations are done, it creates a suitable path by combining the points with minimum cost.

$$F(n) = G(n) + H(n) \tag{6}$$

In Equation 6, n is the previous point of the path, $G(n)$ is the cost of the path from the start point to the adjacent point, and $H(n)$ is the heuristic that estimates the cost of the lowest cost path from n to the target point. In our case, $H(n)$ is the Euclidian distance between corresponding point and target point.

## Experimental Setup

In this section, the software packages and the hardware used are introduced.

### *Software Packages*

In this paper, ROS (Robot Operating System) is used as the middleware for controller development [20]. ROS speeds up controller development considerably due to its readily available libraries. Recently, it has been widely adopted in the robotics literature [21, 22]. ROS provides convenient low-level features such as device control, hardware abstraction, and management of packages, as well as many libraries for mapping and localization, navigation and perception.

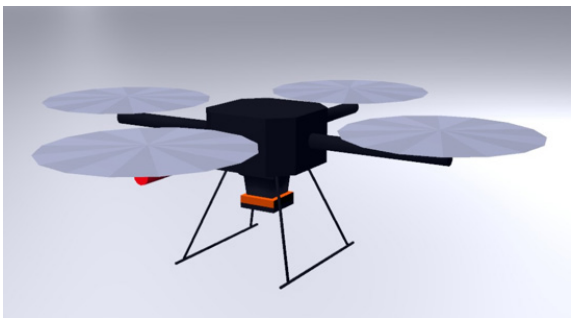


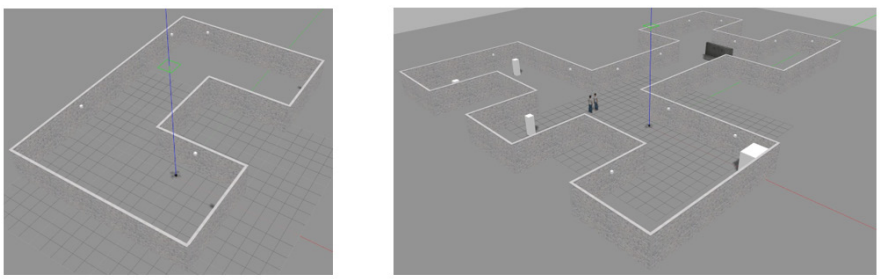
**Figure 1.** Hector Quadrotor [23].

Gazebo is used as the simulation environment. Gazebo is a physics-based 3D simulator used widely in robotics studies. It uses different physics engines such as ODE and Bullet. OGRE is used as the graphics engine. ROS can seamlessly be integrated to Gazebo that eases the development of controllers in Gazebo. In the simulations, a PC with Ubuntu 16.04 LTS(Xenial Xerus) is used running ROS Kinetic Kame distribution and Gazebo 7.1.

### *Quadrotor*

The Hector quadrotor model is used as the quadrotor platform [23]. "Hector SLAM" package including "Hector Mapping, Hector IMU Tools and Hector Nav Msgs" for simultaneous localization and mapping, which is developed especially for indoor environments, are used.

The laser range finder "Hokuyo UTM-30LX LIDAR [24]" is selected and mounted under the UAV. A sonar sensor is used to measure the height of the quadrotor from the ground. An IMU is used to measure the angular velocity and acceleration of the UAV. Lastly, a UWB sensor is used for localization. A tag is mounted on the quadrotor and anchors are placed on the walls of the environment as shown in Fig. 2. A new plugin named "Gazebosensorplugin" that is developed by Barral et al. [15] is adopted for reading the senors. The Hector Quadrotor model is shown with the integrated laser range finder in Fig. 1.



(a) (b)

**Figure 2.** Maps Used for Simulations. UWB anchors are shown with white spheres on the walls. (a) Map 1 [25] (b) Map 2 with Obstacles

**Table 1.** Performance Comparison Table for Map 1 ($\sigma_{Exp}$, $\sigma_{Tar}$ and $\sigma_{W}$ all denote the standard deviation of distance and time for the exploration, target-based and wall following methods, respectively.).

| | *Exploration Algorithm* | *Target-Based Navigation Algorithm* | *Wall Following Algorithm* | $\sigma_{Exp}$ | $\sigma_{Tar}$ | $\sigma_{Wall}$ |
|---|---|---|---|---|---|---|
| *d [m]* | *38* | *36* | *45* | *7,8* | *5,9* | *8,2* |
| *t [s]* | *253* | *247* | *312* | *52* | *40* | *57* |

## EXPERIMENTAL PROCEDURE

In Gazebo, two maps are developed with different complexities. On these maps, the aforementioned navigation algorithms are tested. The size of Map 1 is set to be approximately 280m$^2$ and there are no obstacles present on this map [25]. Similarly, a more challenging map is designed with a size of 850m$^2$ having different sized and shaped obstacles such as standing man, cube and barrier. The corresponding maps are shown in Fig. 2.

All three algorithms are tested on Map 1. However, exploration algorithm failed on Map 2 since it is not smart enough to carry out navigation and mapping on huge maps with obstacles. Therefore, only two algorithms are tested on Map 2. The quadrotor is released from ten different positions in order to check the repeatability of the algorithms. Simulations are completed after the entire map is explored.

### Performance Metrics

All described algorithms perform the same task that is to obtain the map of an unknown indoor environment without colliding with any obstacles. It is required to determine some criteria in order to measure the effectiveness of algorithms in different scenarios. Hence two performance metrics are defined.

#### *Distance*

The distance travelled by the robot is recorded. The smaller the distance is, the more successful the algorithm is. $d$ represents the distance of the robot in meters.

#### *Time*

This criterion is the total time spent throughout the mapping process of an unknown indoor environment. Time is indicated by $t$ and it is measured in seconds.

## RESULTS AND DISCUSSION

### Simulations

The results of the simulations performed in Map 1 are shown in Fig. 3 for the exploration, target-based naviga-

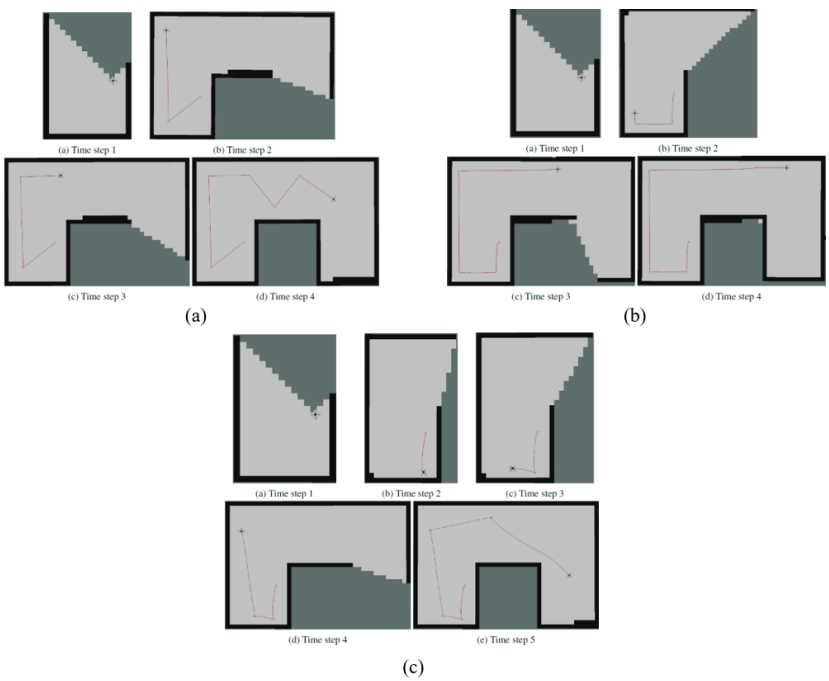



**Figure 3.** Navigation Algorithms for Mapping on Map 1. (a) Navigation with the Target-Based Navigation Algorithm (b) Navigation with the Wall Following Algorithm (c) Navigation with the Exploration Algorithm

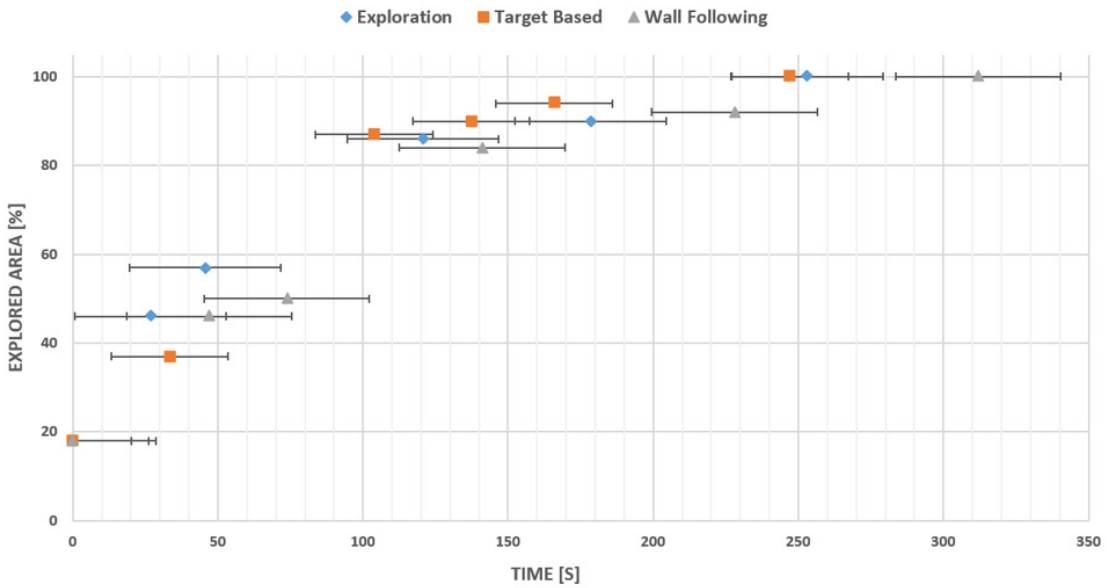



**Figure 4.** Time-Dependent Performance of the Navigation Algorithms on Map 1. Points show the mean and error bars show the variance. Time measurements are taken at some stop points of the corresponding algorithm as shown in Figure 3. For instance, quadrotor stops four times during navigation for the wall following algorithm while it stops five times for the exploration algorithm.

tion and wall following algorithms, respectively. In Fig. 3, the map is explored with five stops for the exploration algorithm while it is explored with six stops for the target-based navigation algorithm. On the other hand, number of stops is four for the wall following algorithm and it is equal to number of corners since wall following algorithm stops if it encounters a wall.

The performance of each algorithm on Map 1 are shown in Fig.s 4, 5 and Table 1 for time and distance, respectively. Target-based navigation algorithm performs better in terms of time and distance travelled when compared to the other two algorithms.

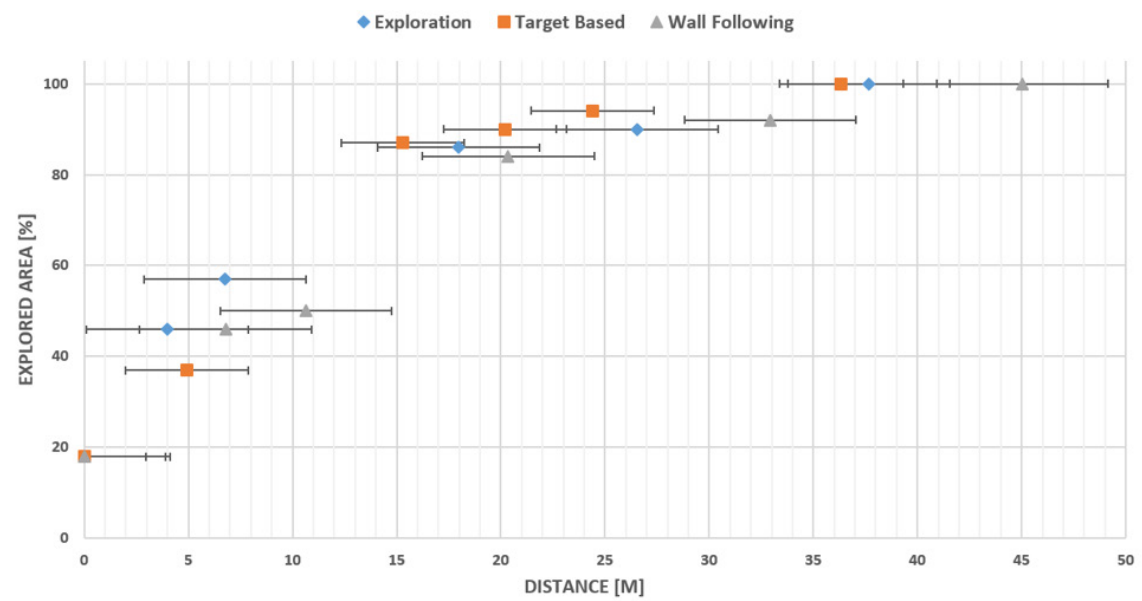



**Figure 5.** Path-Dependent Performance of the Navigation Algorithms on Map 1. Points show the mean and error bars show the variance. Distance measurements are taken at some stop points of the corresponding algorithm as shown in Figure 3. For instance, quadrotor stops four times during navigation for the wall following algorithm while it stops five times for the exploration algorithm.

**Table 2.** Performance Comparison Table for Map 2 with Obstacles ($\sigma_{Tar}$ and $\sigma_{W}$ all denote the standard deviation of distance and time for the target-based and wall following methods, respectively.).

| | *Target-Based Navigation Algorithm* | *Wall Following Algorithm* | $\sigma_{Tar}$ | $\sigma_{Wall}$ |
|---|---|---|---|---|
| *d [m]* | *103* | *182* | *14,1* | *6,68* |
| *t [s]* | *812* | *1393* | *110* | *51* |

The results of the simulations performed in Map 2 with obstacles are shown in Fig. 6 for target-based navigation and wall following algorithms, respectively. In Fig. 6, number of stops decreases for the target-based navigation algorithm due to the presence of obstacles in the environment. The wall following algorithm performs almost the same way as the previous map. However, longer distance is travelled with the wall following algorithm due to the larger size of Map 2.

The performance of each algorithm on Map 2 are shown in Fig.s 7, 8 and Table 2 for time and distance, respectively. Target-based navigation algorithm is far more superior than the wall following algorithm in terms of all performance measures.

## Discussion

In this study, performance of different navigation algorithms on two different maps have been investigated based on time and travelled distance. The exploration algorithm makes the UAV to move to the corner points. It is successful on relatively small environments. However, it fails on the large ones. The simulation results of this algorithm was not shown in Map 2 since it failed on this map. The number of destination points visited is directly proportional to the number of corners as seen in Fig. 3. The algorithm revisits the discovered areas since it does not keep the positions in its memory. Major changes are observed in time and distance travelled values when the initial position of the robot is altered (see Fig.s 4 , 5 and Table 1). This indicates that the algorithm has poor repeatability performance.

Wall following is a well-known algorithm to navigate in indoor environments. Although it is known as a simple and effective algorithm, it has some drawbacks. If the size of the map increases, the distance travelled and the time spent also increase. It explores the map in similar ways. Thus, starting from different points for navigation does not affect the repeatability performance(see Fig. 4, 5, 7, 8 and Tables 1, 2). It does not check whether the map is explored or not since the only reference is wall tracking. It may obtain successful results in the indoor environments with continuous walls. However, it fails if there are multiple continuous walls in the environment.

The target-based navigation algorithm assigns targets and generates obstacle free paths. This algorithm is able to explore both Map 1 and Map 2 for each case. Although the target-based navigation algorithm performed similar to the other two algorithms on Map 1, it outperformed the two on Map 2.

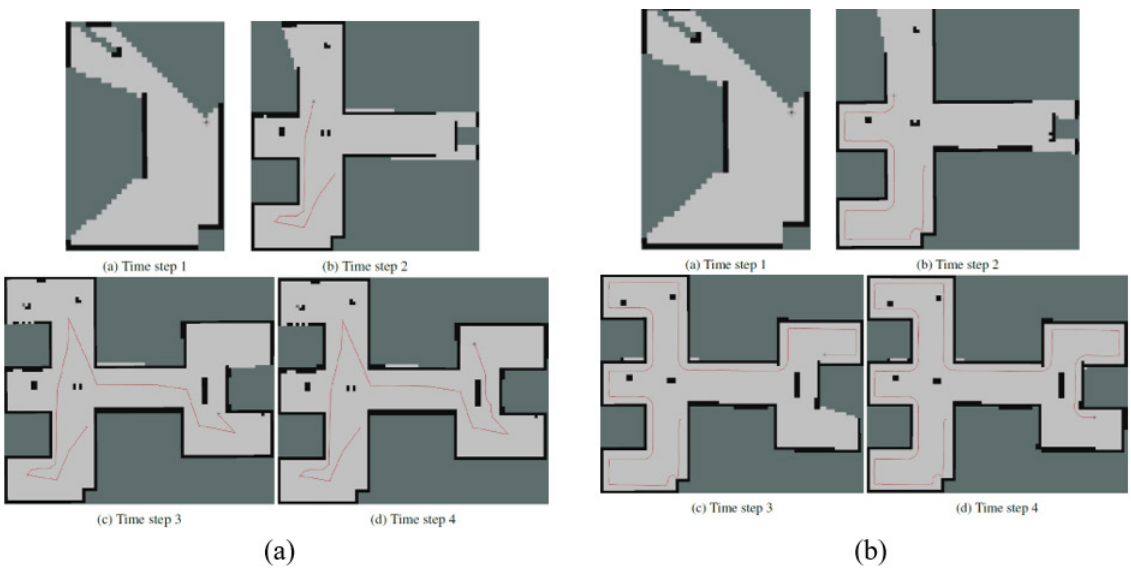



**Figure 6.** Navigation Algorithms for Mapping on Map 2 with Obstacles (a) Navigation with the Target-Based Navigation Algorithm (b) Navigation with the Wall Following Algorithm

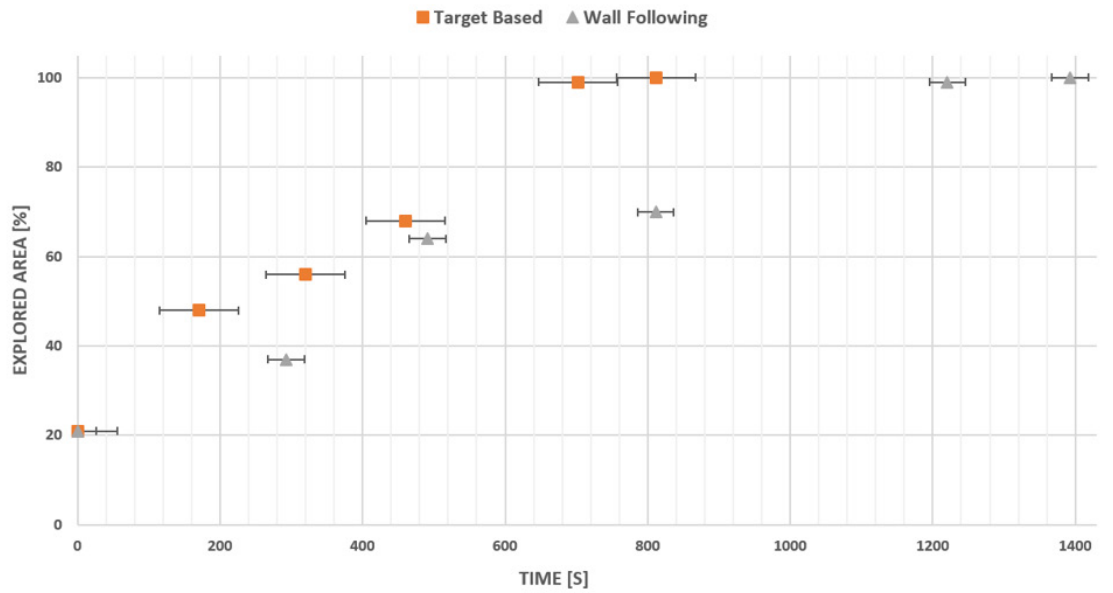



**Figure 7.** Time-Dependent Performance of the Navigation Algorithms on Map 2. Points show the mean and error bars show the variance. Time measurements are taken at some stop points of the corresponding algorithm as shown in Figure 6. For instance, quadrotor stops fourteen times during navigation for the target-based navigation algorithm while it stops twenty two times for the wall following algorithm.

In addition, with the target-based navigation algorithm similar results are obtained regardless of the initial position of the UAV since it has a consistent target generation algorithm as shown in Fig.s 4, 5, 7, 8 and Tables 1, 2. Therefore, repeatability performance is superior compared with the other navigation algorithms.

All in all, the exploration and the wall following algorithms move by using the details such as corners, walls and obstacles in the indoor environment. Therefore, they do not check whether the map has been discovered. On the other hand, the target-based navigation algorithm directly uses the explored areas to select the next destination. This feature guarantees that the map is explored.

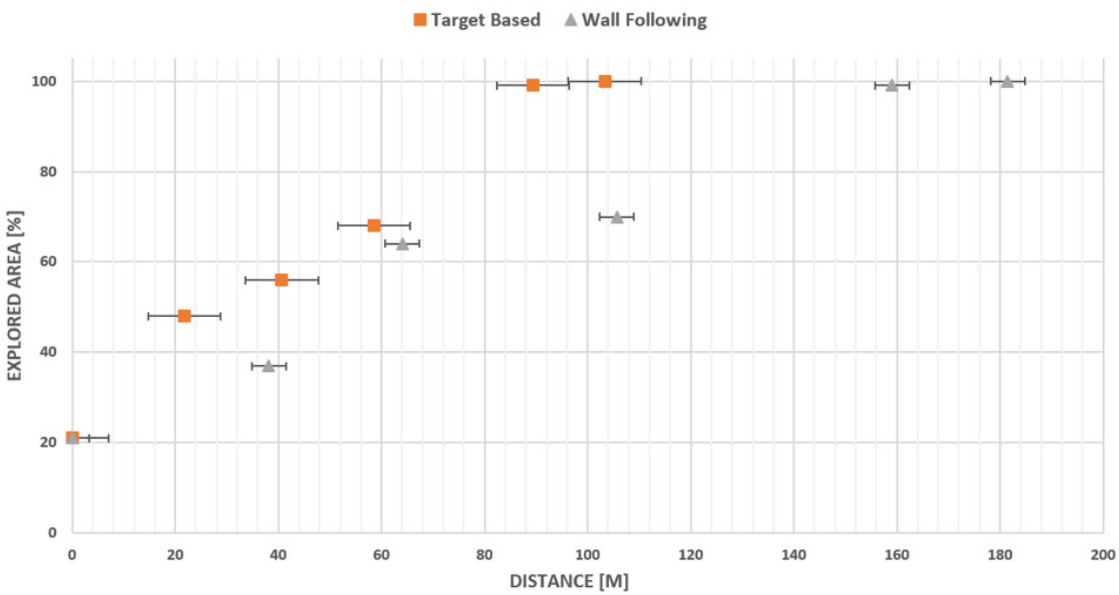



**Figure 8.** Path-Dependent Performance of the Navigation Algorithms on Map 2. Points show the mean and error bars show the variance. Distance measurements are taken at some stop points of the corresponding algorithm as shown in Figure 6. For instance, quadrotor stops fourteen times during navigation for the target-based navigation algorithm while it stops twenty two times for the wall following algorithm.

In addition, comparisons show that the target-based navigation algorithm gets better results than the other navigation algorithms.

## CONCLUSION

In this study, a novel navigation algorithm was developed for a quadrotor in order to obtain the map of unknown indoor environments. Two different navigation algorithms that were presented in the previous studies were used to compare the performance of the proposed novel algorithm using different performance metrics. LIDAR-based SLAM method was used in all algorithms. UWB localization was applied to the exploration algorithm and the novel algorithm by using anchors placed on the walls and a tag mounted on quadrotor. In this way, local positioning system was formed. Local positioning was not used for the wall following algorithm since it does not require localization. Wall following is an old and well-known navigation algorithm. It is suitable for mazelike environments but it can also be used for any indoor environment. However, it performs poorly when time and distance travelled are important. In addition, it does not succeed in indoor environments with large empty spaces. The exploration algorithm uses corners and open spaces to find destination points. Even if it is considered as successful in small indoor environments, it fails in large indoor environments with obstacles. Different than this, the novel algorithm guarantees to obtain the whole map since it directly uses the explored areas by composing a matrix. Navigation system of the wall following and exploration algorithms is not based on explored map. The map of an indoor environment is explored unconsciously by these algorithms since they just use the features of the environment without checking whether the map is explored or not. In addition, novel algorithm may be applied for any robots including aerial robots, ground robots and underwater robots.

The performed simulations showed that the novel algorithm mostly beats the opponents. Exploration algorithm directly failed in large indoor environments including different obstacles. Although wall following algorithm managed to obtain map of indoors, it wasted time and travelled a longer distance. Moreover, repeatability analysis indicates that performance metrics come out with close values for novel algorithm. In other words, navigation with novel algorithm raises similar results regardless of the robot's initial position.

As a future work, the navigation algorithms can be tested in physical environments by setting up the maps in real world. Instead of LIDAR based SLAM method, camera based SLAM method may be used easily by taking advantage of generic novel algorithm. In addition, several quadrotors can be released from different points simultaneously in order to reduce the discovery time of the map and communication among quadrotors can be established using wireless technologies. Furthermore, the novel algorithm can be tested using various aerial, ground and underwater robots.

### References


1. Y. Khosiawan and I. Nielsen, "A system of uav application in indoor environment," Production & Manufacturing Research 4(1), 2–22 (2016).
2. C. Hegde and N. S. Guptha, "Implementation of Mapping Algorithm for SLAM Operation," Ijetae 3(9), 235–238 (2013).
3. A. Araujo, D. Portugal, M. S. Couceiro, et al., "Integrating Arduino-Based Educational Mobile Robots in ROS," Journal of Intelligent and Robotic Systems: Theory and Applications 77(2), 281–298 (2014).
4. H. I. M. A. Omara and K. S. M. Sahari, "Indoor mapping using kinect and ROS," 2015 International Symposium on Agents, Multi-Agent Systems and Robotics, ISAMSR 2015 , 110–116 (2016).
5. N. Johnson, "Vision-Assisted Control of a Hovering Air Vehicle in an Indoor Setting," Engineering and Technology (August) (2008).
6. S. Ahrens, D. Levine, G. Andrews, et al., "Vision-based guidance and control of a hovering vehicle in unknown, gps-denied environments," Proceedings - IEEE International Conference on Robotics and Automation , 2643–2648 (2009).
7. J. F. Roberts, T. S. Stirling, J.-C. Zufferey, et al., "Quadrotor Using Minimal Sensing For Autonomous Indoor Flight," European Micro Air Vehicle Conference and Flight Competition (EMAV2007) (September), 17–21 (2007).
8. M. Kara Mohamed, S. Patra, and A. Lanzon, "Designing simple indoor navigation system for UAVs," 2011 19th Mediterranean Conference on Control and Automation, MED 2011, 1223–1228 (2011).
9. M. Achtelik, J. Williams, M. J. Owen, et al., "Autonomous Navigation and Exploration of a Quadrotor Helicopter in GPS-denied Indoor Environments," First Symposium on Indoor Flight (2009).
10. S. Grzonka, G. Grisetti, and W. Burgard, "A fully autonomous indoor quadrotor," IEEE Transactions on Robotics 28(1), 90–100 (2012).
11. F. Wang, J. Cui, S. K. Phang, et al., "A mono-camera and scanning laser range finder based UAV indoor navigation system," 2013 International Conference on Unmanned Aircraft Systems, ICUAS 2013 - Conference Proceedings , 694–701 (2013).
12. O. Oguejiofor, A. Aniedu, H. Ejiofor, et al., "Trilateration Based localization Algorithm for Wireless Sensor Network," Int. J. Sci. Mod. Eng (10), 21–27 (2013).
13. H. Liu, J. Liu, P. Banerjee, et al., "Survey of Wireless Indoor Positioning Techniques and Systems," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Bucharest, Romania : 1990) 35(1), 39–42 (1991).
14. B. Kempke, P. Pannuto, and P. Dutta, "SurePoint: Exploiting Ultra Wideband Flooding and Diversity to Provide Robust, Scalable, High-Fidelity Indoor Localization," SenSys , 318–319 (2016).
15. V. Barral, P. Su´arez-Casal, C. J. Escudero, et al., "Multi-sensor accurate forklift location and tracking simulation in industrial indoor environments," Electronics (Switzerland) 8(10) (2019).
16. J. R. B. del Rosario, J. G. Sanidad, A. M. Lim, et al., "Modelling and Characterization of a Maze-Solving Mobile Robot Using Wall Follower Algorithm," Applied Mechanics and Materials 446-

447(July), 1245–1249 (2013).
17. E. B. Küçüktabak, M. M. Pelit, Z. O¨ . Orhan, et al., "Kapalı Bir Alanda Basit Bir IHA ile Keşif Metodu Tasarımı Indoor UAV Exploration Method with UWB Localization," TOK , 1–6 (2017).
18. L. Freda and G. Oriolo, "Frontier-Based Probabilistic Strategies for Sensor-Based Exploration," International Conference on Robotics and Automation (April), 3892–3898 (2005).
19. P. E. Hart, N. J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," IEEE Transactions of Systems Science and Cybernetics 4(2), 100–107 (1968).
20. M. Quigley, K. Conley, B. P. Gerkey, et al., "Ros: an open-source robot operating system," in ICRA Workshop on Open Source Software, (2009).
21. M. S. Güzel, V. B. Ajabshir, P. Nattharith, et al., "A novel framework for multi-agent systems using a decentralized strategy," Robotica 37(4), 691–707 (2019).
22. M. S. Güzel, E. C. Gezer, V. B. Ajabshir, et al., "An adaptive pattern formation approach for swarm robots," in 2017 4th International Conference on Electrical and Electronic Engineering (ICEEE), 194–198, IEEE (2017).
23. J. Meyer, A. Sendobry, S. Kohlbrecher, et al., "Comprehensive Simulation of Quadrotor UAVs using ROS and Gazebo," 7628(November) (2012).
24. Hokuyo, "Scanning Rangefinder Distance Data Output/URG-04LX-UG01 Product Details — Hokuyo Automatic Co. Ltd."
25. O. Oral, A. E. Turgut, and K. B. Arıkan, "IHA ile GPS Kullanmadan Kapalı Alanların Haritasının Çıkartılması," ToRK 2019 - Turkiye Robotbilim Konferansı 5(1), 105–111 (2019).