

Decentralized Time-Based Target Searching Algorithm Using Sensor Network Topology Maps

Ashanie Gunathillake
School of Electrical Engineering
and Telecommunication,
University of New South Wales,
Sydney, Australia
NICTA, Eveleigh, NSW 2015, Australia
ashanie@student.unsw.edu.au

Andrey V. Savkin
School of Electrical Engineering
and Telecommunication,
University of New South Wales,
Sydney, Australia
a.savkin@unsw.edu.au

Anura Jayasumana
Department of Electrical and
Computer Engineering,
Colorado State University,
Fort Collins, CO 80523 USA
anura.jayasumana@colostate.edu

Abstract— Real-time target search path planning is a ubiquitous problem present in various areas such as search and rescue, military, and security. In this paper, we consider a search for a static target and a moving target in an unknown environment full of obstacles. The knowledge of target location or direction is obtained by time sets recorded by each sensor node to specify the time instances that the target moved in its sensing region. Our objective is to find the shortest feasible path in a decentralized procedure. The search is conducted by a mobile agent that performs all decision-making procedures based on its local neighborhood sensor information, such as location and the time that the target was seen. A proposed topology map algorithm that does not need any special hardware device embedded with the sensors obtains the sensor locations. The result shows that the proposed algorithm has been able to find the moving target that changes its moving pattern during the search. In addition, the mobile agent avoids the obstacles and searches for the target in an effective way.

Keywords—path planning, packet receiving probability, target search, topological map, wireless sensor network

I. INTRODUCTION

The problem of searching for targets is a vital issue in applications such as emergency rescue, military, security, and natural disasters [1] [2]. These applications use Wireless Sensor Networks (WSNs) to explore the environment and an autonomous robot to chase the target [3]. The WSN estimates the target position and transmits to a sink node to guide the mobile robot towards the target. The sensor observations are routed to a sink node via multiple hops. However, in emergency and military environments, there is a high uncertainty of delivering a packet to the destination via multiple hops due to noise, time delays, and link failures [4]. Thus, we are motivated to propose a decentralizing target-searching algorithm that does not rely on long communication links or on a one central node.

Some of the reasons behind the use of WSNs in target-searching algorithms includes, ease of deployment and low maintenance costs [2]. However, one of the major challenges is to determine the location of the sensor nodes while minimizing hardware costs. To address this issue, numerous techniques have been proposed in the literature that can be divided

into two categories namely; range-based and range-free algorithms. The accuracy of these algorithms depends on factors such as environmental characteristics and anchor distribution. Therefore, we propose our target-searching algorithm that uses topology coordinates of sensor nodes to navigate the autonomous mobile robot. A topology map is an attractive alternative to a physical map, which represents the arrangement of nodes while maintaining the connectivity of nodes. In this paper, we use a topology map obtained by Singular Value Decomposition (SVD) of a Virtual Coordinate System (VCS) proposed in [5].

The search problem proposed in this paper is described as follows. The mobile robot communicates with its local neighborhood and gathers information, such as topology coordinates of sensors and time slots where each sensor node senses the target. The mobile robot navigation search is model as a discrete searching and it terminates when the target is located within robot's sensing region. The search is described by a n by n grid and the agent's next available search location is a function of the agent's current search location. First, we consider a static target-search problem and, secondly, a moving target search problem. The target's movement is independent of the agent's movement and it can be any pattern or a random walk. The objective of our algorithm is to find the shortest path to explore the target, using information from the agent's local neighbors regarding the target's location and direction.

The main contributions of this paper are:

- 1) proposing a decentralized target-searching algorithm,
- 2) proposing an algorithm to use topology map of WSNs where the actual physical distances are hidden, and
- 3) navigating the robot using local neighborhood's topology coordinates and the time sets that the target is detected.

The outline of this paper is as follows. Section II reviews the background of our work. Section II presents the Topology Preserving Map calculation. Section III discusses the details of our proposed algorithm and Section IV presents the results. Finally, Section V presents the conclusions of our work.

II. RELATED WORK

The problem of searching for targets was discussed in different dimensions in the literature. It can be characterized in different approaches as a one-sided or two-sided search, a stationary or moving target, a discrete or continuous time search, and a real-time or off-line search [6] [7]. Initial studies of the problem dealt with a search theory that was mostly concerned with decision problems rather than path construction [6]. Recently, many contributions pay attention on path construction and it is basically carried out as a real-time searching problem, i.e., the path is decided during the search process.

Lau et al. [8] proposed a generalized optimal search path problem based on NP-complete optimal search path problem. The target is moving within a known indoor environment partitioned into interconnected regions. The search path problem is modelled as a discrete search that concerns both cell connectivity and transmit time. Kagan et al. [7] implemented a real-time algorithm based on a probabilistic version of a local search with estimated global distance. The algorithm inputs are the initial target's location probabilities over the considered domain and the transition matrix of a possible target's moves. The decision making at each step of the search applies probabilistic distance estimation and the algorithm objective is to find the path trajectory of the agent that minimizes the average number of search steps.

Korf et al. proposed a Learning Real-Time A* algorithm (LRTA*) for a static target [9] and the Moving Target Search (MTS) algorithm for a target moving around the network [10]. Recently, Kagon et al. [11] proposed the Information Moving Target Search (IMTS) algorithm, which applies a group testing approach in MTS and informational distance measures. Departing from the traditional non-linear model, an integer linear programming formation is proposed in [6]. A team of centrally-controlled homogeneous agents search a grid area to maximize the probability of target detection. The novelty of this approach is that it combines original network representation and approximate problem-solving procedures based on linear programming technology.

III. TOPOLOGY PRESERVING MAP (TPM) FROM VCS

Finding the physical coordinates of sensor nodes is crucial and is yet to be developed. Thus, the target-search problem proposed in this paper uses topology coordinates of WSNs. In this section, we describe a topology map obtained from VCS [5]. VCS characterize a node by a coordinate vector size A , consisting of the shortest hop distance to each of a set of A anchors and preserves information about connectivity, physical layout, and voids. The main reason for choosing this method is that it has a high accuracy and does not depend on any special hardware device.

Let us consider a sensor network with N nodes and A anchors. P , the $N \times A$ matrix containing the VCs of all the nodes. Then, the SVD of P is denoted as;

$$P = U \cdot S \cdot V^T \quad (1)$$

P_{SVD} is an $N \times A$ matrix that describes each node with a new set of A -length coordinate vectors. It can be calculated as;

$$P_{SVD} = P \cdot V \quad (2)$$

Then, the topology coordinates of the nodes are given by $[X, Y]$ and it is the second and third columns of P_{SVD}

$$[X, Y] = [P_{SVD}^{(2)}, P_{SVD}^{(3)}] \quad (3)$$

In the following sections, the topology coordinates of sensor node i is represented by (x_i, y_i) .

IV. TARGET-SEARCH ALGORITHM USING WSN TOPOLOGY MAP

This section describes the decentralized target-search algorithm using WSN topology maps. Time and space are discrete. When the target is detected by a set of sensor nodes $S_t = \{s_1, \dots, s_k\}$, those sensor nodes transmit a message to the sink node. Then, that information is fed to an autonomous mobile robot and it starts to search the target after a t_a time delay. In this paper, we consider static-target and moving-target searches.

A. Static-target searching algorithm

In rescuing applications, the target may become stuck in a place and wait for the rescue team to reach them. Thus, we considered static-target search as our first step. This algorithm is based on A* algorithm [12]. However, the main difference is the proposed algorithm is based on topology information. The main challenge in using a topology map is that the actual distances are hidden. However, the node connectivity information can be used to navigate the robot towards the target.

The objective of the autonomous robot is to find the target with the least possible time by avoiding the obstacles. Here, we consider the eight-neighbour grid search, as shown in Figure 1.

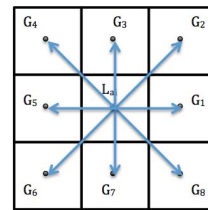


Fig. 1: Eight-neighbour grid

The static target location $L_t = (X_t, Y_t)$ can be calculated using the equations (4) and it is transmitted to the autonomous robot through the sensor network.

$$X_t = \frac{\sum_{s_i \in S_t} x_i}{N_t} \quad \text{and} \quad Y_t = \frac{\sum_{s_i \in S_t} y_i}{N_t} \quad (4)$$

TABLE I: NOTATIONS USED IN THE TEXT

Notation	Description
N	Number of sensor nodes in the network
$A(<< N)$	Number of anchor nodes in the network
s_i	Sensor node i
$[x_i, y_i]$	Topology coordinates of node i
$P_{N \times A}$	Virtual coordinate matrix of the entire network
$L_{t_i} = [X_{t_i}, Y_{t_i}]$	Target location in TPM at time t_i
$L_{a_i} = [X_{a_i}, Y_{a_i}]$	Autonomous robot location in TPM at time t_i
V_t	Speed of the target
V_a	Speed of the autonomous robot
N_{a_i}	Set of local neighbors of autonomous robot at time i
$G_{a_i} = \{G_1, \dots, G_8\}$	Grid points at time t_i
$G_i = [X_{G_i}, Y_{G_i}]$	Topology coordinates of grid point G_i where $i = 1 \dots 8$
$S_{V_a + \varepsilon}$	Set of nodes in $V_a + \varepsilon$ -neighborhood of autonomous robot at time t_i
S_t	Set of nodes that detect the target initially
$N_t = S_t $	Number of nodes that detect the target initially
θ_i	Angle between node i and horizontal axis in TPM
$Z_{s_i}^{t_j}$	Set of time instances of sensor node i captured the target until time t_j
$Z_{G_i}^{t_j}$	Union of time sets of nodes that located in grid G_i at time t_j
Z^{t_j}	Union of $Z_{G_i}^{t_j}$ at time t_j
$M = Z^{t_j} $	Number of elements in Z^{t_j}
Q_m	Weighting factor for the time elements t_m in Z^{t_j}
$Z_{captured}$	Binary matrix with $8 \times M$
W_{nm}	Weighting factor for each element in $Z_{captured}$ matrix
P_{G_i}	Probability of selecting location of grid G_i as autonomous robot's next step

The robot transmits a message to its local neighborhood to request their topology coordinates and transmitting power when the robot makes a stat change. Then, the mobile robot calculates its topology coordinates at time t_i using equation (5). Based on that, the grid locations corresponding to that

state are calculated using equation (6).

$$X_{a_i} = \frac{\sum_{j=1}^{|N_{a_i}|} w_j x_j}{\sum_{j=1}^{|N_{a_i}|} w_j} \quad \text{and} \quad Y_{a_i} = \frac{\sum_{j=1}^{|N_{a_i}|} w_j y_j}{\sum_{j=1}^{|N_{a_i}|} w_j} \quad (5)$$

where, $s_j \in N_{a_i}$ and $w_j = \frac{P_{rx_j}}{P_{tx_j}}$

$$\begin{aligned} X_{G_k} &= X_{a_i} + V_a \cos\{(k-1)\pi/4\} \\ Y_{G_k} &= Y_{a_i} + V_a \sin\{(k-1)\pi/4\} \end{aligned} \quad (6)$$

where, $k = 1, \dots, 8$

Once the grid layout is calculated by the robot, its next step is to perform a search to find the shortest path to the target position. For that, our cost function is

$$F(n) = G(n) + H(n) \quad (7)$$

where, $G(n)$ is the cost of moving from the initial step to the next step on the grid and $H(n)$ is the cost of moving from a grid point to the target's location.

For the $H(n)$ calculation, we use the Euclidean distance in the topology map. However, the actual length of the path cannot be calculated because obstacles can be in the way of the target. Thus, $H(n)$ is an approximate value. The robot calculates $F(n)$ value for all the grid locations and chooses the grid point that has minimum $F(n)$ value as it's next state.

B. Moving-target searching algorithm

In military and security applications, the target moves around the environment. Thus, we enhance our algorithm to a moving-target search algorithm, which increases the reliability and real-time responsiveness by decentralizing. The reason we cannot use the previous cost function directly is because it does not provide any information about the direction of the target movements. Hence, we modified the cost function with the time instances that each node captured the target.

The initial target location L_{t_0} , autonomous robot location at each time L_{a_i} , and grid locations G_i are calculated by using equations (4), (5), and (6) respectively. The autonomous robot transmits a message to its $V_a + \varepsilon$ -neighborhood¹ to request their time sets $Z_{s_i}^{t_j}$. ε is a value within 0 and $V_a/2$. The topology coordinates of its local neighborhood nodes cannot be used to find the nodes located within the region of $V_a + \varepsilon$, because V_a is the speed of the robot on a physical map. Therefore, the robot's transmission power needs to be adjusted to transmit a message to its $V_a + \varepsilon$ -neighborhood. Then, the nodes that responded $S_{V_a + \varepsilon}$ are labeled as $c_1, \dots, c_8$ based on their angle to the x-axis θ_i as in equation (8). These labels indicates where the nodes are located in the grid, e.g.: a node labeled as c_1 is located in the G_1 grid.

$$c_k = (2k-3)\pi/8 < \theta_i < (2k-1)\pi/8 \quad k = 1, \dots, 8 \quad (8)$$

¹nodes located within a circle centered at the robot's location with a radius of $V_a + \varepsilon$

The $Z_{G_i}^{t_j}$ is defined based on the labels of sensor nodes as in equation (9). To calculate the next state of the robot, $Z_{G_i}^{t_j}$ sets of all grids need to be compared. One may think, the grid having the latest time instance should be the next state of the robot. However, there may be one or more grids having the latest time instance in their set. Hence, we need to find a method to choose the best grid point for the next robot state. Even though the latest time instance is in the time set of the grid, that doesn't mean the target is closer to that grid. The reason is because, at the latest time instance, the target can be located at the edge of a node's sensing region or at the center of a node's sensing region. Therefore, the actual closest node is the node that has the target in the center of its sensing region. That can be obtained by considering the past time instances. Hence, a weighting factor is calculated for each time instance, which increases from oldest time instance to the latest. The calculation of that weighting factor is as follows.

$$Z_{G_i}^{t_j} = \{t_k : t_k \in Z_{s_i}^{t_j} \text{ and label of } s_i = c_i, \text{ where } s_i \in S_{V_a+\varepsilon}\} \quad (9)$$

Z^{t_j} is the union of all grid time sets and is defined as $Z^{t_j} = \{t_k : t_k \in Z_{G_i}^{t_j} \text{ where } i = 1 \dots 8\}$. A binary matrix $Z_{captured}$ is defined with dimension of $8 \times M$ where M is the number of elements in Z^{t_j} . The $Z_{captured}$ rows correspond to the grids and columns corresponds to the elements of Z^{t_j} . The elements of the matrix follows the rule that,

$$Z_{captured}(n, m) = 1, \text{ if } t_m \in Z_{G_i}^{t_j} \\ Z_{captured}(n, m) = 0, \text{ else}$$

To find the next state of the autonomous robot, W_{nm} is calculated as,

$$W_{nm} = \begin{cases} Q(m) & \text{if } Z_{captured}(n, m) = 1 \\ 1 - Q(m) & \text{if } Z_{captured}(n, m) = 0 \end{cases}$$

where

$$Q(m) := \frac{t_m - \min(Z^{t_j}) + 1}{\max(Z^{t_j}) - \min(Z^{t_j}) + 1}; \quad t_m \in Z^{t_j} \quad (10)$$

Then P_{G_i} , the probability of selecting G_i grid point as the next state of the robot is defined in equation (11). Finally, the maximum value of P_{G_i} is selected and the corresponding grid location is chosen as the next state of the robot trajectory.

$$P_{G_i} = W_{i1} \dots W_{iM} \quad (11)$$

V. SIMULATION RESULT

The performance of the proposed algorithm is evaluated using MATLAB simulation software. Two different network setups are used for the evaluation. Figure 2(a) shows a $30m \times 30m$ sparse network with 700 nodes and Figure 2(b) shows a 496 nodes $30m \times 30m$ circular shape network with three obstacles. The anchor nodes are selected by using ENS method proposed in [13]. The TPM of Figure 2(a) and Figure 2(b) networks are shown in Figure 2(c) and Figure 2(d) respectively. We assumed that the networks are deployed in suburban area, and path loss and shadowing components are selected based on [14] [15].

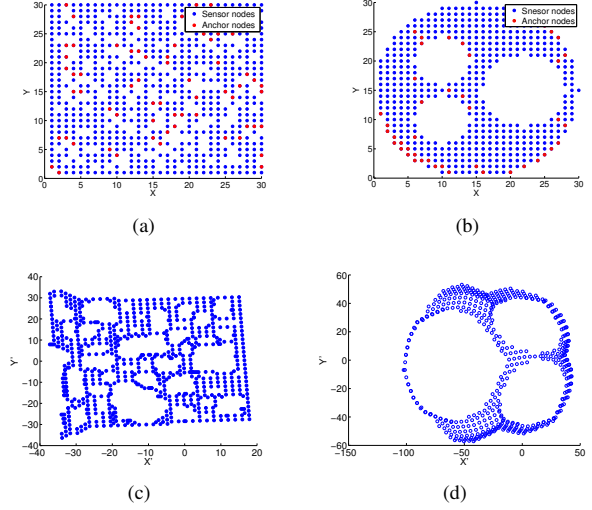


Fig. 2: (a) Sparse network with 700 nodes (b) Circular network with 496 nodes and three obstacles (c) TPM of sparse network and (d) TPM of circular network

We evaluate the proposed static-target search and moving-target search algorithms with P-G algorithm proposed in [2]. The P-G algorithm is based on pseudo-gradient approach that required communication hop count and Received Signal Strength (RSS) in the node neighborhood. To simulate the communication link, we use the equation (12) and the simulation parameters are shown in Table II.

$$P_{rx_i} = P_{tx_j} - 10\varepsilon \log d_{ij} - L_{ob_i} + v_i \quad (12)$$

where, the received signal strength at node i is P_{rx_i} , the transmitted signal strength of the signal at node j is P_{tx_j} , the path-loss exponent is ε , the distance between node i and node j is d_{ij} , L_{ob_i} is the loss due to signal absorption from obstacles exist in the line of sight of node i and j , and the logarithm of shadowing component on node i at is v_i . The shadowing values are selected from a normal distribution with zero mean parameter and standard deviation parameter described as in Table II. The absorption coefficient and the thickness of the obstacle medium, which signal traverses are α and d_o respectively. Then L_{ob_i} can be calculated as,

$$L_{ob_i} = \sum_{k=1}^n 10\alpha d_o \log(e) \quad (13)$$

where, e is the exponent and n is the number of obstacles exist in between node i and node j .

To perform a fair evaluation between two algorithms, we need to convert the result to one domain. Therefore the autonomous robot trajectory in topology map is mapped to the physical map using Procrustes Analysis. Procrustes Analysis can align two sets of data points. It calculates the scaling factor, rotation angle and shift between the two sets [16]. The transformation factors are calculated for each step of the robot based on its local neighborhood nodes coordinates in physical and topology maps.

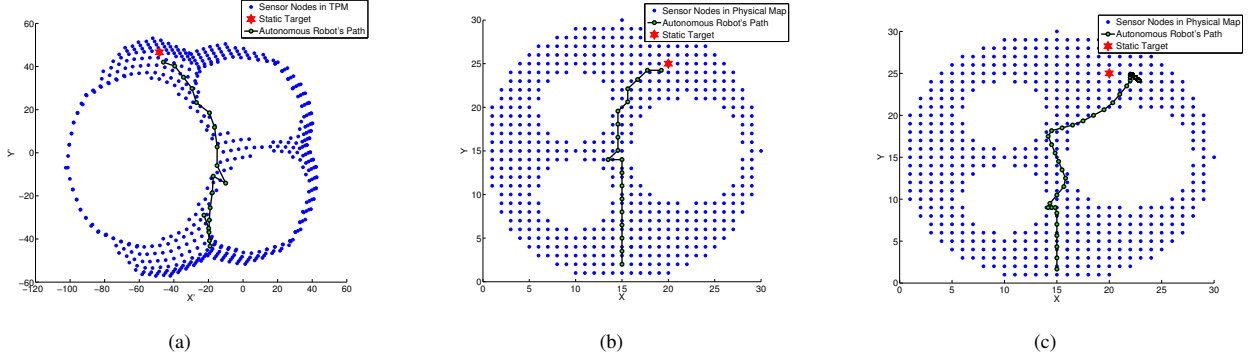


Fig. 3: Static-target search using (a) Proposed algorithm in topology map (b) Proposed algorithm in physical map and (c) P-G algorithm in physical map

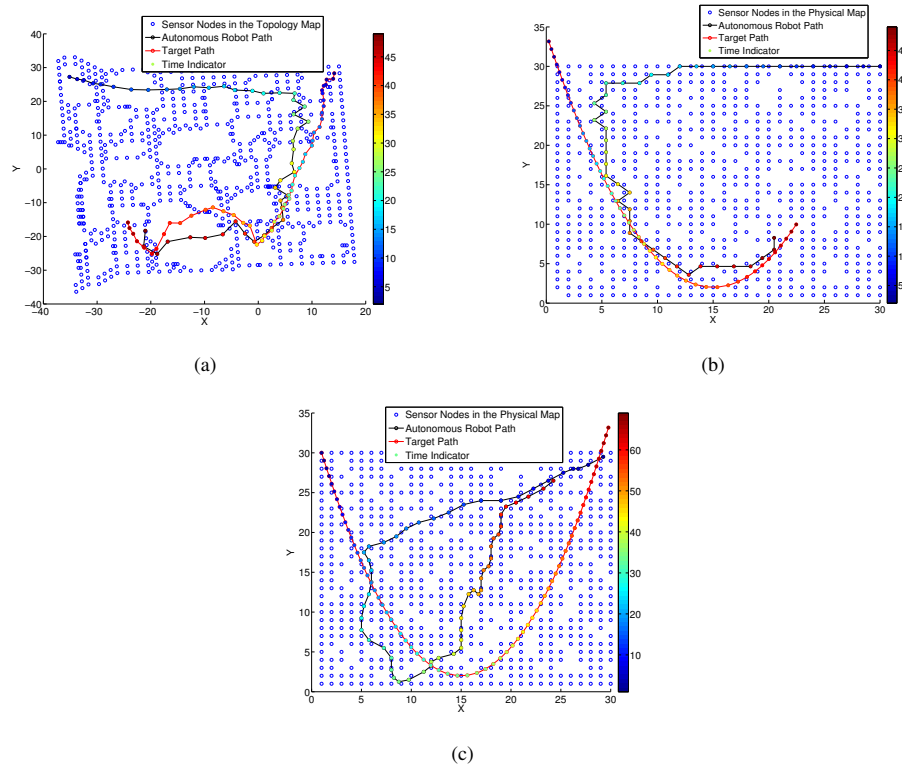


Fig. 4: Searching a moving-target that walk along a parabola, using (a) Proposed algorithm in topology map (b) Proposed algorithm in physical map and (c) P-G algorithm in physical map

Let consider step i have n number of nodes in it's local neighborhood. Then X and Y are the topological and physical coordinate matrices with length of $n \times 2$ respectively. Thus the transformation factors of step i can be estimated by equation (14).

$$Y = bXT + c; \quad (14)$$

where b is the scaling factor, T is the rotation angle and c is the shift value. When the transformation factors are calculate, physical coordinates of step i can be determined using equation

(14) where X and Y are topological coordinates and physical coordinates of step i .

A. Static-target search algorithm evaluation

To evaluate the performance of static-target search, a network full with obstacles is chose as shown in Figure 2(b). The static target is located at (25, 25) coordinates in the physical map and the autonomous robot starts to search the target from (15, 1)-physical coordinates. The results of the proposed algorithm and P-G algorithm are shown in Figure 3. Figure

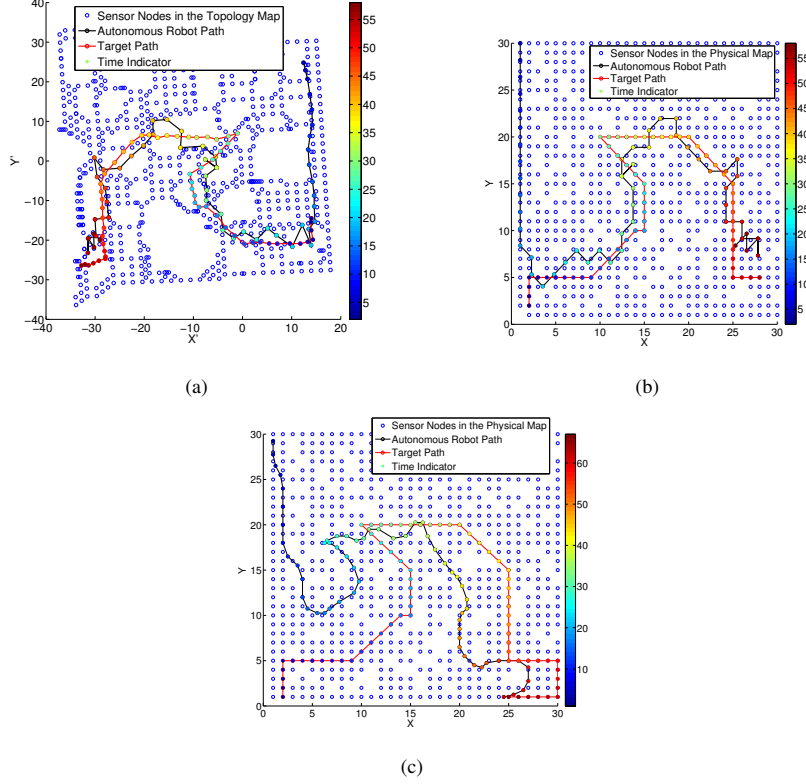


Fig. 5: Searching a moving-target that has a random walk using (a) Proposed algorithm in topology map (b) Proposed algorithm in physical map and (c) P-G algorithm in physical map

TABLE II: SIMULATION PARAMETERS

Parameter	Value
Sensitivity P_{rx}	-80dB
Path-loss exponent ε	2.7
Standard deviation of shadowing σ	9.6
Radius of sensing region of a sensor node	2m
Target speed	$1ms^{-1}$
Autonomous robot speed	$1.5ms^{-1}$

3(a) shows the autonomous robot search path of the proposed algorithm in the topology map and Figure 3(b) shows the transformed path to the physical map. The robot takes 20s to find the target. The robot search path using the P-G algorithm is shown in Figure 3(c) and it takes 25s to reach the target, which is larger than the proposed algorithm. Moreover, it can be seen that the robot search path using the proposed algorithm is straight towards the target than the P-G algorithm search path.

B. Moving-target search algorithm evaluation

To evaluate the performance of moving-target search, networks shown in Figure 2(a) and Figure 2(b) are used. We simulate three cases with different target moving patterns.

Case 1: Parabolic target moving pattern in sparse network starting at (0,33)-physical coordinates. The autonomous robot starts to search the target after 5s of detecting the target and search starts from (30,30)-physical coordinates

Case 2: Random walk of a target in sparse network starting at (2,1)-physical coordinates. The autonomous robot starts to search the target after 5s of detecting the target and search starts from (1,30)-physical coordinates

Case 3: Random walk of a target in circular network with obstacles starting at (1,16)-physical coordinates. The autonomous robot starts to search the target after 5s of detecting the target and search starts from (3,9)-physical coordinates

Figure 5-6 show the results of three cases. In those three figures, sub-figure (a) shows the proposed algorithm's autonomous robot search path in the topology map and sub-figure (b) shows the mapped path in the physical map. The robot search path using the P-G algorithm is shown in sub-figures (c). The three figures illustrate that the proposed algorithm

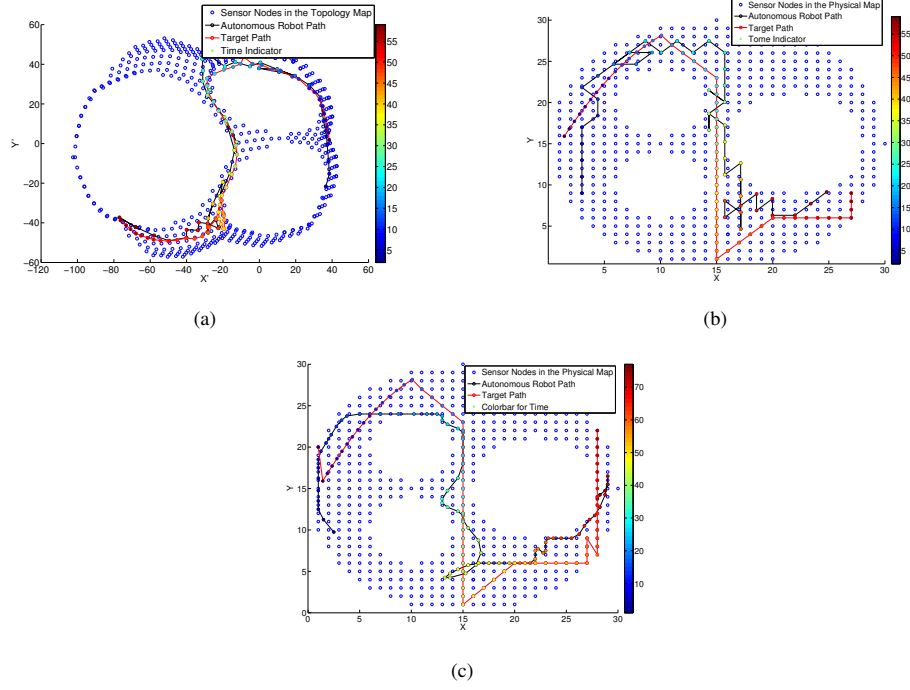


Fig. 6: Searching a moving-target that has a random walk in an obstacle environment using (a) Proposed algorithm in topology map (b) Proposed algorithm in physical map and (c) P-G algorithm in physical map

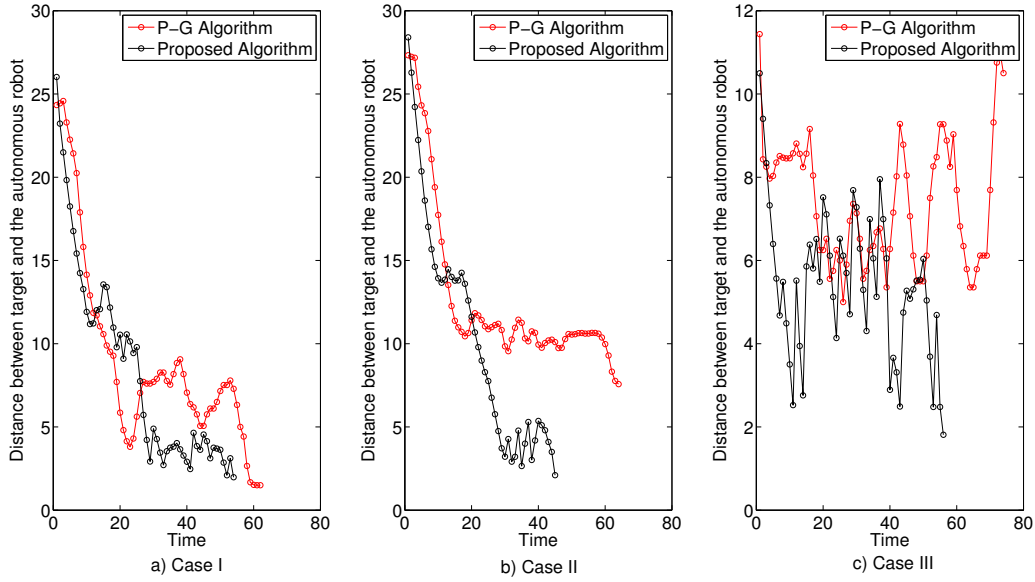


Fig. 7: The distances between the target and the robot at each time

follows the robot path closely to the target trajectory and capture the target quickly than the P-G algorithm. To extract more features, we used time indicators along the paths of the robot and the target, to interpret where they are located at each time. Moreover, we plot the distances between target and the

robot at each time and it is shown in Figure 7. It can be seen that, the robot move quickly to the target's direction using the proposed algorithm. In Figure 7-c the robot takes a longer time compared to other two cases, because target changes it direction very quickly and the environment is full with

TABLE III: TIME REQUIRED TO CAPTURE THE MOVING-TARGET IN FIGURE 4 - 6

	Case I	Case II	Case III
Proposed Algorithm	45s	55s	57s
P-G Algorithm	-	67s	-

obstacles. However with the proposed algorithm, the robot has been able to capture the target before leaving the environment.

Moreover, the time taken to capture the moving target is shown in Table III. In case I and III, P-G algorithm could not be able to capture the target with $2ms^{-1}$ speed before it leaves the network. The main reason is, P-G algorithm decision making is depending only on the current position of the target. However we checked the time required to capture the target by increasing the robot speed to $3ms^{-1}$, and it was 61s and 72s for Case I and Case III respectively. In the proposed algorithm, we use the past time instances with a weighting factor for the decision making process. As a result, the proposed algorithm has been able to capture the target with lesser time compared to other algorithms. Also, the proposed algorithm consumes less energy and have less complexity compared to P-G algorithm. In P-G algorithm the pseudo-gradient is calculated in each step of the robot movement, which required to get the hop count of all nodes in the network. This consumes much energy while increasing the traffic of the network. However in the proposed method, robot communicates with it's local neighborhood that required short length communication link. Hence it gives real-time result with less energy consumption.

VI. CONCLUSION

This paper proposed static-target and moving-target search decentralized algorithms that use the topology coordinates of WSN. In literature, the target-search algorithms use the physical coordinates of sensor nodes. However, finding the physical coordinates with less cost and without using any special hardware devices is challenging and has yet to be developed. Hence, we used topology coordinates of sensors, which is an alternative to the physical coordinates and accurately represents the network layout.

The proposed algorithm can be used in unknown environments containing obstacles to search the target. The knowledge of the target location or direction is obtained by time sets where each sensor node captured the target when it was moving in those nodes'sensing regions. The autonomous mobile robot performs all decision-making procedures based on its local neighborhood information. Thus, it has a real-time output with less energy consumption and traffic in the network.

The performance of the proposed algorithm is compared with the recently proposed P-G algorithm. In the simulations, the target's movement is independent of agent's movement and we simulate a parabolic pattern and a random walk of the target. From the result, it can be seen that the proposed algorithm has outperformed and it follows the target path very closely to capture the target quickly. The main reason for that

is it uses the past and present information with a weighting factor for the decision-making.

ACKNOWLEDGMENT

This research is supported in part by Data61|CSIRO Australia.

REFERENCES

- [1] D. A. Grundel, "Searching for a moving target: optimal path planning," in *Proceedings. 2005 IEEE Networking, Sensing and Control, 2005.*, pp. 867–872, March 2005.
- [2] N. Deshpande, E. Grant, and T. C. Henderson, "Target localization and autonomous navigation using wireless sensor networks 2014;a pseudogradient algorithm approach," *IEEE Systems Journal*, vol. 8, pp. 93–103, March 2014.
- [3] S. Beyme and C. Leung, "Rollout algorithms for wireless sensor network-assisted target search," *IEEE Sensors Journal*, vol. 15, pp. 3835–3845, July 2015.
- [4] S. Gagan, D. M. Weeraddana, and A. Gunathillake, "Sensor network based adaptable system architecture for emergency situations," *Lecture Notes on Information Theory*, vol. 2, pp. 85–91, March 2014.
- [5] D. Dhanapala and A. Jayasumana, "Topology preserving maps ;extracting layout maps of wireless sensor networks from virtual coordinates," *IEEE/ACM Transactions on Networking.*, vol. 22, pp. 784–797, June 2014.
- [6] N. Lo, J. Berger, and M. Noel, "Toward optimizing static target search path planning," in *2012 IEEE Symposium on Computational Intelligence for Security and Defence Applications*, pp. 1–7, July 2012.
- [7] E. Kagan, G. Goren, and I. Ben-Gal, "Probabilistic double-distance algorithm of search after static or moving target by autonomous mobile agent," in *IEEE 26th Convention of Electrical and Electronics Engineers in Israel (IEEEI), 2010*, pp. 000160–000164, Nov 2010.
- [8] H. Lau, S. Huang, and G. Dissanayake, "Probabilistic search for a moving target in an indoor environment," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3393–3398, Oct 2006.
- [9] X. Sun, S. Koenig, and W. Yeoh, "Real-time adaptive a*" in *In proceedings of the international joint conference on autonomous agents and multiagent systems*, pp. 281–288, 2008.
- [10] T. Ishida and R. E. Korf, "Moving-target search: a real-time search for changing goals," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 609–619, Jun 1995.
- [11] E. Kagan and I. Ben-Gal, "Moving target search algorithm with informational distance measures," in *The Open Applied Informatics Journal*, vol. 6, pp. 1–10, 2013.
- [12] B. M. ElHalawany, H. M. Abdel-Kader, A. TagEldeen, A. E. Elsayed, and Z. B. Nossair, "Modified a* algorithm for safer mobile robot navigation," in *Proceedings of International Conference on Modelling, Identification Control (ICMIC), 2013*, pp. 74–78, Aug 2013.
- [13] D. C. Dhanapala and A. P. Jayasumana, "Anchor selection and topology preserving maps in wsns 2014; a directional virtual coordinate based approach," in *IEEE 36th Conference on Local Computer Networks (LCN), 2011*, pp. 571–579, Oct 2011.
- [14] S. Seidel, T. Rappaport, S. Jain, M. Lord, and R. Singh, "Path loss, scattering and multipath delay statistics in four european cities for digital cellular and microcellular radiotelephone," *IEEE Transactions on Vehicular Technology.*, vol. 40, pp. 721–730, Nov 1991.
- [15] V. Erceg, L. Greenstein, S. Tjandra, S. Parkoff, A. Gupta, B. Kulic, A. Julius, and R. Bianchi, "An empirically based path loss model for wireless channels in suburban environments," *IEEE Journal on Selected Areas in Communications.*, vol. 17, pp. 1205–1211, Jul 1999.
- [16] V. D. M. Nhat, N. Vo, S. Challa, and S. Lee, "Nonmetric mds for sensor localization," in *3rd International Symposium on Wireless Pervasive Computing, 2008. ISWPC 2008.*, pp. 396–400, May 2008.