Indonesian Journal of Electrical Engineering and Computer Science
Vol. 15, No. 2, August 2019, pp. 743~749
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v15.i2.pp743-749

A review on graph search algorithms for optimal energy efficient path planning for an unmanned air vehicle

Sanjoy Kumar Debnath[1], Rosli Omar[2], Nor Badariyah Abdul Latip[3], Shasha Shelyna[4], Elia Nadira[5], Che Ku Nor Che Ku Melor[6], Tapan Kumar Chakraborty[7], Elango Natarajan[8]
[1,2,3,4,5,6]Faculty of Electrical and Electronics Engineering, Universiti Tun Hussein Onn Malaysia, Malaysia
[7]Department of Electrical and Electronics Engineering, University of Asia Pacific, Bangladesh
[8]Faculty of Engineering, Technology and Built Environment, UCSI University, Malaysia

Article Info

Article history:

Received Oct 12, 2018
Revised Feb 15, 2019
Accepted Mar 29, 2019

Keywords:

Combinatorial method
Graph search algorithm
Optimal energy efficient
Path planning
Unmanned air vehicle

ABSTRACT

Unmanned Air Vehicle (UAV) has attracted attention in recent years in conducting missions for longer time with higher levels of autonomy. For the enhanced autonomous characteristic of UAV, path planning is one of the crucial issues. Current researches on the graph search algorithms under combinatorial method are mainly reviewed in this paper by keeping focus on the comprehensive surveys of its properties for path planning. The outcome is a pen picture of their assumptions and drawbacks.

Corresponding Author:

Rosli Omar,
Faculty of Electrical & Electronic Engineering,
Universiti Tun Hussein Onn Malaysia,
Parit Raja, Batu Pahat-86400, Johor, Malaysia.
Email: roslio@uthm.edu.my

1. INTRODUCTION

A UAV or robot without path planning does not have the abilities to complete the surveillance and rescue tasks [1] like 2001 World Trade Centre collapse [2], Hurricane Katrina in 2005 [3], and the 2011 Tohoku tsunami and earthquake [4]. Therefore, path planning is necessary for it to aid human operators in dangerous circumstances, such as natural disasters and hazardous areas [5]. There are mainly three kinds of path planning techniques namely combinatorial, sample based and biologically inspired and they are illustrated in Figure1.

In sampling based method, rapidly-exploring random tree (RRT) does not always provide the optimal result and probabilistic roadmap (PRM) is expensive without any guarantee to find the path. Genetic Algorithm (GA) is considered as a biologically inspired method and it is currently utilized in energy efficient path planning. But it also cannot guarantee the achievement of the optimal path because local minima may occur in narrow environments.

Moreover, GA is computationally expensive and practically not complete. Moreover, when the fitness functions are not well-defined, it usually tries to converge along local optima of the problem avoiding its global optimum. A self-adaptive differential evolution (SDE) algorithm can eliminate the required manual tuning of control features [6]. Particle Swarm Optimization (PSO) has real time effect but it can easily fall into local optima in many optimization problems. Furthermore, there is no general convergence theory applicable to PSO in practice and its convergence time is also uncertain for multidimensional problems [34]. Ant Colony Optimization (ACO) does a blind search and thus, it is not suitable for energy saving path planning due to the lack of optimal result. Apart from very slow and very high cost functions, Simulated Annealing (SA) is also not capable of finding the optimal path.

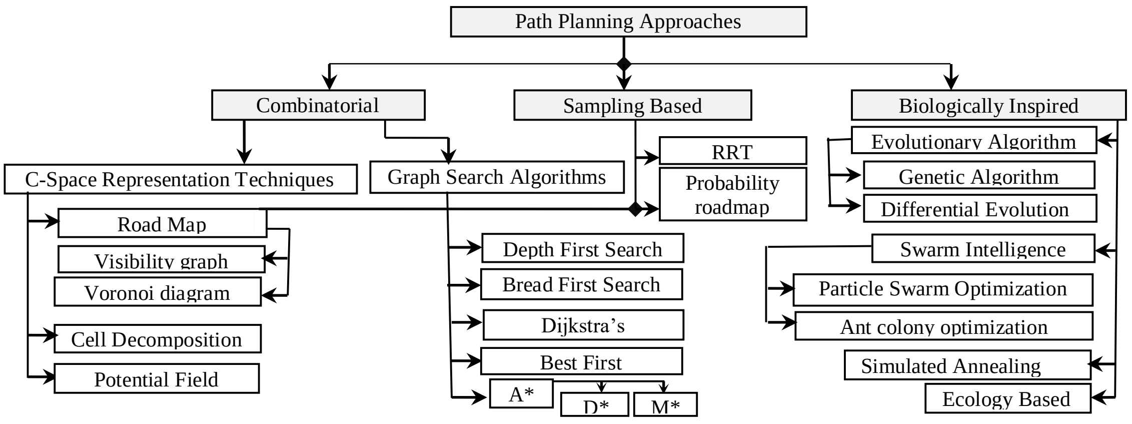

Figure 1. Classification of path planning approaches

Under the combinatorial method, potential field (PF) sometimes does not find the goal because of local minima issue. The necessary adjustments are required for cell decomposition (CD) as per the situation; e.g., in exact CD, the cells are not predefined, but they are selected based on the location and shape of the obstacles within the configuration space (C-space) [7]. Visibility graph (VG) is more energy efficient than voronoi diagram (VD) in combinatorial method under roadmap technique because the latter fails to create the shortest path [8,33] for which there is a high plausibility of wastage in energy consumption and cost.

UAV has limited energy or power or fuel and this restrains it from a longer time flight. Therefore, it is important for a UAV to adopt a path planning algorithm ensuring that the traversed path must be collision-free and optimal in terms of path length and energy efficiency. The most common problem in path planning that a UAV has to come across is a set of obstacles through which it needs to fly in between the given starting and target points [8] within the specified area. These obstacles may not be fixed at one location and can pop up during the fly. Configuration space (C-space) is a most commonly used technique for path planning to denote the position of obstacles and the information of free space in a given area. It provides detailed position information of all points in the system and is the space for all configurations. It assumes the UAV as a point and adds the area of the obstacles so that the path planning can be done more efficiently. C-space is obtained by adding the UAV radius while gliding along the edge of the obstacles and the border of the search space. An illustration of a C-space for a circular UAV is shown in Figure 2.

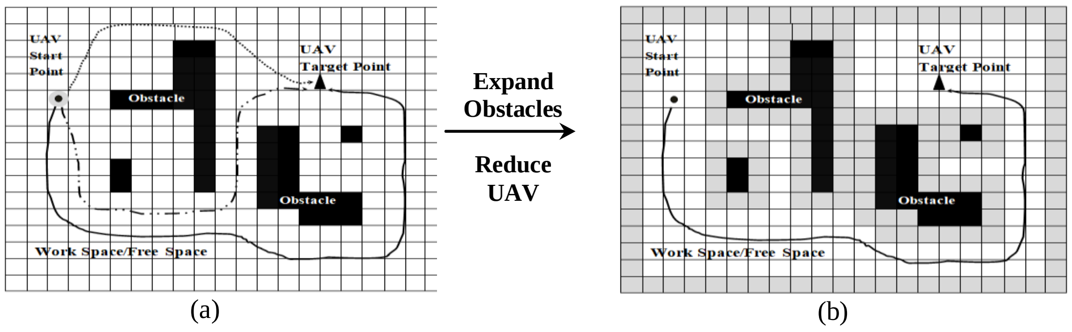

Figure 2. Configuration space for a UAV

2. GRAPH SEARCH ALGORITHM

Graph search algorithms have been used extensively in past studies for energy efficient path planning. It generally determines a path from starting to target points by checking some nodes/states. Without any existing path, it will report failure. Several graph search methods are discussed below.

2.1. Depth-First Search (DFS)

DFS moves towards the goal as quickly as possible and searches a path till getting the dead end. DFS may miss large portions of the workspace [7, 9] since it tries to search several paths at a time before completing one path. It can be applied to find a path among many possible paths. However, DFS is an uninformed search since the cost function is not used in deciding the suitable direction and in estimating that how far is the target point from the current node.DFS may be slightly faster in case it picks the leaf node path that contains the required node, but it is not complete. When numerous solutions are in the tree where everyone is at a comparable `depth', then there is a chance to miss a larger part of the tree from exploring. Conversely, there is a chance for it to stuck in the lengthy blind alleys, whereas fewer steps solution path exists and hence, it is not the best solution. When the depth values in the search are fixed, it prevents the above issue. But this method is not much effective. DFS is good in selecting one solution among many possibilities without any prior knowledge and not suitable when only one or the shortest solution exits. In DFS, the required memory is linear against the search graph making it advantageous. It keeps the record of the nodes in the 'current' path leading to less memory requirement for tree search. It is an exhaustive and systematic search method that utilizes every node in the finite search-space.

DFS incorporated Genetic algorithm to discover the optimal processing sequence of features of a part (PSFP) that reduces the feature transitions' energy consumption by 28.60 % [11]. A smaller search space was explored faster with reduced cost by another extended depth-first search (EDFS) algorithm [32].

2.2. Breadth-First Search

Moore introduced the Breadth-first Search algorithm in 1957 [12]. It is a systematic search algorithm because it first expanded the shallow nodes by searching all the next level nodes of the path and then it takes the next step. However, like DFS, Breadth-first Search is an uninformed search to find the shortest path in first attempt. It is applicable in limited solutions that use comparatively minimum steps [13].

Breadth-first Search algorithm uses more memory and traverses all nodes. It always provides first solution in finding shortest path or determines a path with minimum steps without getting stuck in any blind alleys. The main feature of Breadth-first Search is that when all the graph's edges have no weight or same weight, the shortest path lies within the first visited node and the source node. This algorithm is complete if one exists. It is also a systematic and exhaustive search technique that eventually tries all the nodes in the search space (if it is finite!). Breadth-first Search is faster than DFS when the required information is closer to the root of the beginning of the search. However, the total speed depends on the information storage procedure.

The memory requirement of Breadth-first Search is high because it saves each level record [14] and this is its main limitation. It is mainly used to find the shortest path between any two nodes in a graph such as road networks, computer networks; social networks (e.g. Facebook) [15].

2.3. Best-First Search (BFS)

Best-first search is classified as a heuristic search algorithm, which uses the distance from a current node with respect to the target point in order to find a path in a graph. BFS is an instance of graph search algorithm in which a node is selected for expansion based on evaluation function f (n) [17-18]. A heuristic is used for making a guess of such distance. Traditionally, the node which is the lowest evaluation is selected for the explanation because the evaluation measures the distance to the goal. The current node's heuristic cost is compared with all other nodes' cost to determine the path in a graph, when BFS is applied. It by-passes few branches to the search tree leading to the non-assurance about the discovery of the shortest path. Best first search is the combination of breadth first search and DFS algorithms and it is employed through a priority queue in a general search framework where the data structure maintains the fringe in the ascending order of values.

BFS algorithm is often referred greedy algorithm because it quickly attacks the most desirable path as soon as its heuristic weight becomes the most desirable. However, its searching action is lesser than Dijkstra's algorithm. BFS utilizes a priority queue Like Dijkstra's to gather the list of nodes. The start node is assigned in the priority queue as first. With expansion, the neighboring nodes are directly connected by having their respective total heuristic cost and they are also stored in the priority queue. The nodes having minimum cost are expanded in the succeeding level. Other nodes in the queue then added. Till the reaching of the target point, this process is repeated.

Best-first search algorithm by-passes few branches to the search tree leading to the non-assurance about the discovery of the shortest path. Certain cost is considered for the goal from the current state. Due to the heuristic function, few paths look good to be continued. Sometimes, it covers more distance than our consideration. On the other hand, BFS guides up to the goal by acquiring the domain information. The time complexity of BFS is much less than Breadth first search. The Best first search allows us to switch between paths by gaining the benefits of both breadths first and depth first search. Games and web crawlers applied BFS and its advanced variants [21] as a path-finding algorithm. For example, in the game the enemy agent can be assigned by it for finding the player's location. Some games divide up the terrain into "tiles" which can either be blocked or unblocked. In such cases, the search algorithm treats each tile as a node, with the neighboring unblocked tiles being successor nodes, and the goal node being the destination tile [22].

2.4. Dijkstra's algorithmEdsger

Dijkstra's introduced this systematic search algorithm in 1959 [23] to find an optimal path in between the initial and all other points in the graph as per the costs associated with traversal. The priority queue saves the cost of the nodes which is non-negative. Dijkstra's algorithm visits all the nodes within the graph starting from an initial point and it is complete if a solution exists. It does not calculate the distance between each node and the target in optimal cases, if no prior knowledge of the graph exists. Dijkstra's algorithm does a blind search for which lots of time are required and wastage of necessary resources occur while processing.

All the nodes in a provided weighted graph are searched in this method based on their respective distance from the starting point in an increasing order. The nearest node from the initial point is determined by the priority queue that operates in a monotonic way. In discrete event simulation, events are prioritized by the times at which they happen and again are extracted monotonically. Likewise, the events where the point of interest is crossed by a sweep line are prioritized in computational geometry's sweep line algorithms by the coordinates of the crossed point and they are done in monotonic order that occurs in the BSF branch and bound [21-22].

Prior information about target node is not required in Dijkstra's and hence, this makes it an uninformed algorithm. It is applicable in an environment with multiple nodes without any priori about the closest node. At each step, it chooses the edges having smallest cost and occasionally, does not need to investigate all the edges. Since it is more general, therefore it is open not only to acyclic graphs, but to others also. Usually it searches a huge territory in the graph and thus, applicable in geographical maps, e.g, Google map. In this algorithm, edges that have positive weights are stored in a priority queue and they are referred by the distance between the locations. In IP routing also Dijkstra's can be used so that in the beginning the Open shortest Path can be found, such as in telephone network.

Palossi et al. implemented Dijkstra's algorithm for an energy-efficient shortest trajectory planning [26]. For multi-rotor UAVs they utilized Intel's HD Graphics accelerator that is generally applied in such missions where human operator's increasing audio-visual competency is required, such as rescue mission. The proposed method saved about 98 % of required energy by the sequential version. Anjali Jain et al. implemented the modified Dijkstra's algorithm which can also increase the performance of finding out the shortest route with minimum iterations and saves cost as well as energy [27]. J. T. Economou et al. also used this algorithm to propose 4-zone strategy that was effective in a multi-scenario method to map the UAV mission. The achieved result revealed that the optimum path was dependent on the minimization of the gross propulsion energy [28].

2.5. A* Search Algorithm

In 1968, Hart [29] proposed a heuristic search method A*. The forward cost function, h(n) along with backward cost function, g(n) in between the source and current nodes are used in this algorithm to calculate the total cost function f(n) as is expressed below combining both the Best-first search and Dijkstra's algorithm:

$f(n)=g(n) + h(n)$

Generally it ignores the obstacle in between the current and target node to find the heuristic value that is is a straight line distance [1]. A* seems to be more like Dijkstra's algorithm when the backward cost is prevailing resulting in the shortest path between the source and target nodes. But A* becomes Dijkstra's Algorithm for zero heuristic and when the searches are for longer time in extreme cases. Conversely, A* seems to be BFS with dominant forward cost or heuristic weighting to producing a fast non-guaranteed shortest path. Nevertheless, it is not possible to calculate in advance the true heuristic value [9]. Total number of nodes to be discovered by A* are reduced by the heuristic function where the start node is saved in the priority queue as the first node with which other neighboring nodes are directly connected. Then the start node is associated with their corresponding total cost and stored in the priority queue. In next step, the least cost neighboring node expands and other adjacent nodes that are not in the queue are added. Till the target point comes in the queue, this process repeats itself. When the heuristic value is lesser than the actual value A* algorithm provides optimal value by giving guaranty to generate a least cost path in between the starting and target points. If a solution exists, then A* is complete only when the time and memory are unlimited.

The A* search algorithm is an example of a best-first search and known as B*. Best-first algorithms are often used for path finding in the combinatorial search which is a best-first graph search algorithm that finds the least-cost path from a given initial node to any goal node [19-20].

A* takes much processing time and decreases the work speed. To overcome this problem Guruji et al. proposed a modified A* algorithm [30] which reduced the processing time at least by 65%. Thus it can be concluded that the modified A* algorithm is better in terms of processing time trading off a little cost of path length and can be applicable for fast processing applications. Sudhakara et al. developed Enhanced A* algorithm which provided a much shorter time of travel as compared to that of the modified A* Algorithm [31]. Therefore, the Enhanced A* Algorithm can accommodate a provision for generating a better control of the robot and pave the way for expediting an effective contrivance for exploring path planning in robotics. A route was found in [16] with either minimum fuel consumption or shortest traveling time by implementing vehicular-ad-hoc-network- (VANET-) based A∗ (VBA∗) route planning algorithm. A* is best regarding the shortest distance, but the algorithm need a longer computational time. The algorithm is not appropriate to the situation for sequential tasks are assigned for robot [24]. The path planning with energy considerations insight were generated by weighted A* search to obtain the global navigation strategy [25].

3. RESULTS AND ANALYSIS

DFS is good to pick up only solution among many possibilities without caring about exactly which one. It may be less appropriate when there is only one solution, or if the shortest one is needed. DFS is good because a solution can be found without computing all nodes

Breadth-first search is suitable for limited available solutions that use a comparatively small number of steps. Its exceptional property finds the shortest path from the source node up to the node that it visited first time when all the graph’s edges are either un-weighted or having similar weight. Breadth-first search is complete if one exists. Breadth-first search is good because it does not get trapped in dead ends.

BFS algorithm does not assure to discover the shortest path because it bypasses some branches in the search tree. It is a greedy search which is not complete and optimal.

Dijkstra’s algorithm is systematic search algorithm and gives shortest path between two nodes. In optimal cases, where there is no prior knowledge of the graph, it cannot estimate the distance between each node and the target. Usually, a large area is covered in the graph by Dijkstra’s due to its selection of edges with minimum cost at every step and thus, it is significant for the situation having multiple target nodes without any prior knowledge of the closest one.

A* is not very optimal because it needs to be executed a number of times for each target node to get them all. A* expands on a node only if it seems promising. It only aims to reach the target from the current node at the earliest and does not attempt to reach any other node. A* is complete because it always finds a path if one exists. By modifying the used heuristics and node’s evaluation tactics of A*, other path-finding algorithm can be developed. Dijkstra and the BFS can be simulated in this way. Table 1 compares the discussed graph search methods in terms of path optimality, computation time, real time capability, memory usage, safety and completeness.

Table. 1 Comparison of Graph Search Methods

	Graph Search Method					
Method	Optimal Path	Computational Time	Real-time	Memory	Safety	Completeness
Depth-First Search	×	×	×	√	×	√
Breadth-First Search	×	×	×	√	×	√
Dijkstra’s	√	×	×	×	√	√
Best First	×	√	√	√	×	√
A*	×	√	√	√	√	√

4. CONCLUSION

Several path planning algorithms were implemented by earlier researchers and now it is multidimensional. The main objectives are to make a path planning efficient depends on (i) minimal computation time, (ii) optimal collision-free path, (iii) energy efficiency, and (iv) completeness. This work concentrated on graph search algorithms considering the objectives of UAV’s mission where energy efficiency and their nature of motion, advantages, and drawbacks were taken care. According to the obtained results such as safety, completeness and computation time, an optimization can be done to make it an energy-efficient path planning algorithm.

ACKNOWLEDGEMENTS

This research supported by Universiti Tun Hussein Onn Malaysia (UTHM) and funded by TIER-1 Vot H131.

REFERENCES

[1] E. Mueggler, M. Faessler, F. Fontana, and D. Scaramuzza. Aerial-guided navigation of a ground robot among movable obstacles. In Safety, Security, and Rescue Robotics (SSRR), 2014, *IEEE International Symposium on, 2014.*

[2] Murphy, Robin R. "Trial by fire [rescue robots]." *IEEE Robotics & Automation Magazine* 11.3 (2004): 50-61.

[3] R. R. Murphy, S. Tadokoro, D. Nardi, A. Jaco, P. Fiorini, H. Choset, and A. M. Erkmen. Search and rescue robotics. In B. Siciliano and K. Oussama, editors, Springer Handbook of Robotics, pages 1151–1173. *Springer Verlag,* 2008.

[4] E. Guizzo. Japan earthquake: Robots help survivors for *http://spectrum.ieee.org/automaton/robotics/industrial-robots/japan-earthquake-robots-help-search-for-survivors,* 2011

[5] J. Nikolic, M. Burri, J. Rehder, S. Leutenegger, C. Huerzeler and R. Siegwart. A uav system for inspection of industrial facilities. *In Aerospace Conference, 2013 IEEE*, pages 1, 8, March 2013.

[6] Binitha, S., and S. Siva Sathya. "A survey of bio inspired optimization algorithms." *International Journal of Soft Computing and Engineering* 2.2 (2012): 137-151.

[7] J. Giesbrecht and Defence R&D Canada. Path planning for unmanned ground vehicles. Technical Memorandum DRDC Suffield TM 2004-272, (2004).

[8] Omar, R. and Gu, D.W., 2009," Visibility line based methods for UAV path planning". *In ICCAS-SICE, August 2009 (pp. 3176-3181). IEEE.*

[9] S. M. LaValle. 2006, *Planning Algorithms*, Cambridge University Press.

[10] Debnath, S.K., Omar, R. and Latip, N.B.A., "A Review on Energy Efficient Path Planning Algorithms for Unmanned Air Vehicles". *In Computational Science and Technology Springer, Singapore*. 2019; pp. 523-532.

[11] Hu, Luoke, et al. "Minimising the energy consumption of tool change and tool path of machining by sequencing the features." Energy 147 (2018): 390-402.

[12] E. F. Moore. 1957, the shortest path through a maze. Proceedings of an International Symposium on the Theory of Switching. Cambridge: *Harvard University Press,* pages 285-292

[13] G. Dudek and M. Jenkin. 2000, Computational principles of mobile robotics. *Cambridge University Press, Cambridge, UK.*

[14] Zafar, Aqsa, Krishna Kant Agrawal, and Wg Cdr Anil Kumar. "Analysis of Multiple Shortest Path Finding Algorithms in Novel Gaming Scenario." *Intelligent Communication, Control and Devices*. Springer, Singapore, 2018. 1267-1274.

[15] https://www.quora.com/What-are-some-real-life-examples-of-Breadth-and-Depth-First-Search;10:29 PM, 10th May2018

[16] Chang, Ing-Chau, et al. "A VANET-Based A* Route Planning Algorithm for Travelling Time-and Energy-Efficient GPS Navigation App*." International Journal of Distributed Sensor Networks* 9.7 (2013): 794521.

[17] Pearl, J. Heuristics: Intelligent Search Strategies for Computer Problem Solving. *Addison-Wesley, 1984*. p. 48.

[18] Russell, Stuart J.; Norvig, Peter (2003), Artificial Intelligence: A Modern Approach (2nd ed.), Upper Saddle River, New Jersey: *Prentice Hall, ISBN 0-13-790395-2.* pp. 94 and 95 (note 3).

[19] Berliner, Hans. "The B* tree search algorithm: A best-first proof procedure." *Readings in Artificial Intelligence*. 1981. 79-87.

[20] Pearl, Judea. *"Heuristics: intelligent search strategies for computer problem solving."* (1984).

[21] Mehlhorn, Kurt; Sanders, Peter (2008). Algorithms and Data Structures: The Basic Toolbox (PDF). *Springer.*

[22] https://en.wikipedia.org/wiki/Monotone_priority_queue

[23] E. W. Dijkstra. 1959, A note on two problems in connexion with graphs. *In Numerische Mathematik* 1, pages 269-271.

[24] Korkmaz, Mehmet, and Akif Durdu. "Comparison of optimal path planning algorithms." Advanced Trends in *Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), 2018 14th International Conference on. IEEE, 2018.*

[25] Gupta, Gaurav, and Ashish Dutta. "Trajectory generation and step planning of a 12 DoF biped robot on uneven surface." *Robotica (2018):* 1-26.

[26] Daniele Palossi, Michele Furci, Roberto Naldi. 2016, *An Energy-Efficient Parallel Algorithm for Real-Time Near-Optimal UAV Path Planning*. CF'16, May 16-19, Como, Italy.

[27] Jain, Anjali, U. Datta, and Neelam Joshi. 2016, "Implemented Modification in Dijkstra's Algorithm to Find the Shortest Path for 'N'Nodes with Constraint*." International Journal of Scientific Engineering and Applied Science* 2.2: 420-426.

[28] J.T. Economou, G Kladis, A Tsourdos, B.A. White. 2007, UAV Optimum Energy Assignment using Dijkstra's Algorithm. *In Proceedings of the European Control Conference* 2007, Kos, Greece, July 2-5.

[29] P. Hart. 1968, A formal basis for the heuristic determination of minimum cost paths. *In IEEE Transactions on Systems Science and Cybernetics,* pages 100-107.

[30] Guruji, Akshay Kumar, Himansh Agarwal, and D. K. Parsediya. 2016, "Time-efficient A* Algorithm for Robot Path Planning*." Procedia Technology* 23: 144-149.

[31] Sudhakara, Priyanka, and Velappa Ganapathy. 2016, "Trajectory Planning of a Mobile Robot using Enhanced A-Star Algorithm." *Indian Journal of Science and Technology* 9.41.

[32] Li, Chao, and Maomi Ueno. "An extended depth- first search algorithm for optimal triangulation of Bayesian networks." *International Journal of Approximate Reasoning* 80 (2017): 294-312.

[33] Latip, N.B.A., Omar, R. and Debnath, S.K. "Optimal Path Planning using Equilateral Spaces Oriented Visibility Graph Method." *International Journal of Electrical and Computer Engineering (IJECE),* 7(6), pp.3046-3051, 2017.

[34] Lim, Wei Hong, Nor Ashidi Mat Isa, Sew Sun Tiang, Teng Hwang Tan, Elango Natarajan, Chin Hong Wong, and Jing Rui Tang. "A Self-Adaptive Topologically Connected-Based Particle Swarm Optimization." *IEEE Access* 6 (2018): 65347-65366.

BIOGRAPHIES OF AUTHORS

Sanjoy Kumar Debnath is a PhD scholar in Faculty of Electrical & Electronic Engineering in the Universiti Tun Hussein Onn Malaysia (UTHM). He received his Masters of Engineering from Universiti Teknologi Malaysia in 2014. He joined a research on "Optimal Energy Efficient Path Planning for an Unmanned Air Vehicle (UAV) in Obstacle-Rich Environment" in 2015 at UTHM under the Office of Research, Innovation, Commercialization, and Consultancy Management.

Dr.Rosli Omar currently is a lecturer at the Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia. He received his PhD in engineering from University of Leicester, United Kingdom in 2012. His research interests are in robotic engineering, autonomous system and system identification.

Nor Badariyah Abdul Latip, received her bachelor in Electronic Engineering followed by Master in Electrical Engineering from Universiti Tun Hussein Onn Malaysia (UTHM) in 2015 and 2018 respectively. Her research interests are in robotic path planning, control system and medical electronic. She is currently working in a multinational company.

Dr. Tapan Kumar Chakraborty received his Bachelor of Science in Electrical and Electronic Engineering degree from Bangladesh University of Engineering and Technology, Dhaka in 1984. He completed the M. Engg degree in Electrical Engineering at the University of Roorkee, India in 1988. He obtained his Ph. D degree in Electrical and computer Engineering from Kanazawa University, Japan in 1998. He served as lecturer, assistant professor, associate professor and professor in the department of Electrical and Electronic Engineering at Dhaka University of Engineering and Technology from October,1988 to May, 2005. He served as the professor and chairman in the department of Electrical and Computer Engineering at the Presidency University, Dhaka from May, 2005 to September, 2016. From October, 2007 to May, 2015, he served as the Dean of the School of Engineering at Presidency University, Dhaka. In September, 2016, he joined the department of Electrical and Electronic Engineering as a professor at the University of Asia Pacific, Dhaka. He has over twenty seven research papers to his credit in various journals and conferences of national and international repute. He is also currently serving as Director of the Institute of Energy, Environment, Research and Development (IEERD) at the University of Asia Pacific in addition to his normal duties. His fields of interests are electronic materials and devices, phase change memory, electronic circuits and power electronics. He is a Fellow of the Institution of Engineers, Bangladesh and a member of the IEEE.

Elango Natarajan is a Chartered Mechanical Engineer (CEng.), who specialized in Mechanical Engineering Design, CAE and Soft Robotics. He obtained doctoral degree in Mechanical Engineering from Anna University, Chennai, India in 2010. He worked as a Post doctoral research fellow in UTM, Skudai, Malaysia in 2013. He has served for engineering colleges/university for about 19+ years in various academic positions. He has been teaching Mechanical Engineering courses since 1999 and he has gained extensive knowledge and experience in Engineering design and Stress analysis, CAE, Vibration, Statics, Soft robotics and polymer composites. He is a active member of IET and IEEE professional bodies.

Che Ku Nor Che Ku Melor is a PhD scholar in Faculty of Electrical & Electronic Engineering in the Universiti Tun Hussein Onn Malaysia (UTHM). She received his Masters of Electrical Engineering from UTHM in 2012.Her research interest in robotic path planning.

Elia Nadira is a PHD in Faculty of Electrical and Electronic in the UTHM. She received her masters of Engineering from UTHM in 2012. She joined a research on "Efficient Cooperative path planning approach based on Potential field method using UAVs" in, 2013 at UTHM.

Shasha Shelyna is a PHD student under the Faculty of Electrical and Electronic in the UTHM. She received her Masters of Engineering from UTHM in 2015. She joined the research on robot navigation through obstacles using SLAM in 2014 at UTHM.