Bidirectional Meta Heuristic Search Method For Robot Navigation

Mohammed Abdul Al-Gileel
Zeyad Taha Yaseen
Baghdad College of Economic Sciences University

Abstract

In this paper, we propose a hybrid search method to improve the path planner that operates on 2D robot work space with some obstacles that is expressed Regularity grids. The optimal path is planned by using bidirectional search method which combine the meta-heuristic behavior of D* algorithm and the power of evaluation function associated with A* search strategy. This method reduced the unnecessary cost of path planning and made the robot path planner more smoothly .

Keywords: Robot navigation , meta heuristic , D* algorithm, A* algorithm

طريقة تنقيبيه بحثية ذو اتجاهين للسيطره على الملاحة الروبوتية

تم في هذا البحث استعراض طريقة بحث هجينة ذو اتجاهين لتحسين مسارات الروبوتات التي تعمل في بيئة ذات بعدين والتي تحوي مجموعة من العوائق والممثلة على شكل شبكة من المواقع المنتظمة . تدمج الطريقة المقترحة سلوك التنقيب عالي المستوى الموجود في خوارزمية *D مع القدرة التنقيبية الموجوده في *A . اثبتت الاختبارات التي طبقت على بيئتين مختلفتين ان الطريقة المقترحة قللت من الكلف غير الضرورية وجعل مسار الروبوت اكثر وضوح و مقبولية .

1. Introduction

The development of technology has brought great conveniences to human beings. Autonomous robotics plays a vital part in accomplishing useful tasks without human intervention in real world and unmodified environments that have not been specifically engineered for the robot [1]. A variety field of study is inter-related with robotics. They are such as Control, Vision, Mechanics, Electrical and Electronics (EE), Information Technology (IT) and Artificial Intelligence (AI) where Vision, Control and Artificial Intelligence (AI) are strongly related with mobile robot navigation. The inter-relationship between robotics and other fields of study is shown in Fig - 1 . The successful use of an Autonomous Mobile Robot (AMR) is strongly related to its controller. Most autonomous robots use sensors to correspond with the environment [2]. The controller of these autonomous robots acts according to the information received by the sensor. Control of robotics systems has many challenging problems especially in unstructured environments. The majority of robots operates in structured environments and performs simple repetitive tasks. However, operations in unstructured environments require robots to perform more complex tasks that may involve autonomous navigation, compliant motion, operational redundancy, fault tolerance, end -effector positioning, and flexing of links and joints. These operations often have to be completed in dynamic, remote, hazardous or, in general, uncertain surroundings. Humans cannot have exact and complete prior knowledge of these environments as many details are usually unknown, the position of people and objects cannot be predicted a priori, passageways may be blocked, and so on. Furthermore, knowledge acquired through sensing is affected by uncertainty and imprecision. The quality of sensor information is influenced by the sensor noise, the limited field of view, the conditions of observation, and the inherent difficulty of the perceptual interpretation process. Thus, major challenge of autonomous robotics is the large amounts of uncertainty that characterizes real world environments. Current research in robotics aims to build an autonomous and intelligent robot which can plan its motion in a dynamic and uncertain environment. According to [3] , robot is a programmable machine that imitates the action or appearance of an intellectual creature such as a human being or an animal. This electro-mechanical device can perform autonomous or preprogrammed task. These autonomous or preprogrammed tasks can be any of the dangerous jobs, dirty jobs and boring jobs that human . A number of software has been developed to control actions and movements of mobile robots to accomplish those preprogrammed or autonomous tasks. All these software needs effective motion planning. Motion planning is often designed to optimize specific performance criteria and to satisfy constraints on the robot ‘s motion [4] . Typical performance criteria are minimum time to arrive at a milestone and minimum control effort. Typical constraints in the motion planning are obstacle avoidance and maximum robot velocity fig(2) . But apparently, due to the lack of in-depth consideration from all aspects or the misunderstanding which occurs during the requirements capturing processes, some problems were found in these developed systems. More and more problems have arisen in the development of the robots boasting. Among them, the problem of getting the accurate and shortest path in the robot navigation is quite apparent in the development of robotics. Mobile robot is one of the developed advanced technologies which assist humans in various purposes and tasks in many application fields such as education, service, industry, exploration, transportation, and more. mobile robot is mainly used for transporting the materials from one point to another. It is known as Automated Guided Vehicle (AGV) . By using AGV in the industry, the cost can be minimized and efficiency can be increased. Navigation is one of the important purposes that uses mobile robot [5] . Mobile robot navigation is the technique to guide the mobile robot moving from the starting position towards the desired goal, along a desired path or accomplishing the desired purposes where static, dynamic, known and unknown environment is involved . The environment of mobile robot navigation is distinguished by variable terrain, a set of distinct objects or certain objects such as obstacles, milestones, and landmarks [6] that may block the movement of the robot in reaching the desired destination or accomplishing the desired purposes. The environment for navigation could be known or unknown with moving and static obstacles. The environment with static obstacles is called static environment meanwhile the environment with moving obstacles that causes the locations of the obstacles change with time is called dynamic environment. Unknown environment is the environment where the locations of the obstacles are unknown to the mobile robot while known environment is the environment where the locations of the obstacles are known to the mobile robot before navigation is carried out. [7].

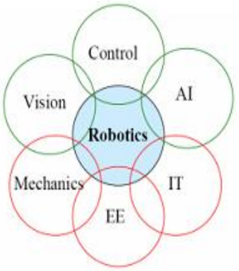
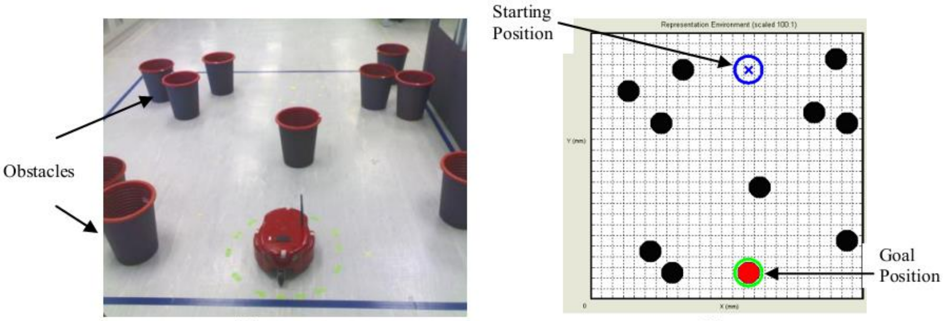
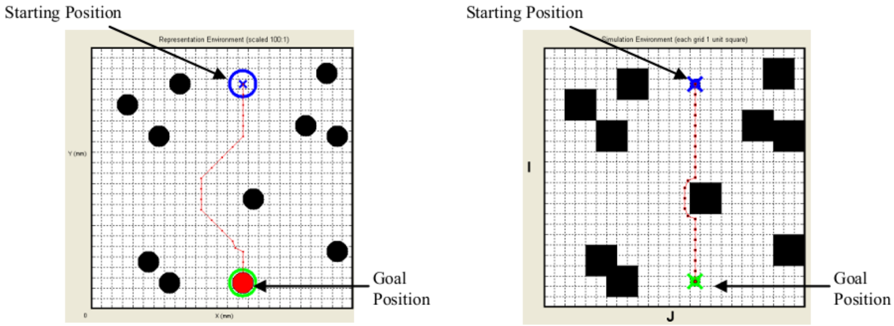


Figure 1: Fields Related to Robotics (Bräunl, 2003)



(a) (b)

Figur2 (a) : actual environment (b) : representation environment


Starting Position
Representation Environment (scaled 100:1)
Y (mm)
X (mm)
Goal Position
Starting Position
Simulation Environment (each grid 1 unit square)
I
J
Goal Position

(a) (b)

Figure3 : 2D work space with obstacles dark spots and squares

2- A* Algorithm

Is an early developed popular graph search algorithm which finds the shortest path from a given initial start node to the goal node. A* Algorithm uses distance plus path cost function to determine the shortest path to the goal node [8] . In A* Algorithm, node or square notation is used rather than coordinate because a map is divided into small grids or squares and nodes represent the center point of each grid . Dynamic A* Algorithm, is developed by [9] .

2.1 **A* operators**

- **OPEN list :** contains nodes currently expanded and ready to be searched.
- CLOSED list : contains nodes already searched until goal state is reached. CLOSED list contains the solution path.
- h(n) : heuristic function which measure the estimated distance from any node n to goal state.
- g(n) : cost function which measure the actual distance from start state to any node n.
- f(n): evaluation function which is the of two distance mentioned by g(n) and h(n) .

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

3- **D* Algorithm**

Is one of most popular goal-directed navigation algorithm that is widely used for mobile robot navigation in unknown environment [9]. D* Algorithm is a reverse or backward searching method and it is able to re -plan from current position when there is a new obstacle blocking the path. The algorithm works iteratively selecting a node from the OPEN list and evaluating it . It then propagate the node's changes to all of the neighboring nodes and places them on the OPEN list. This propagation process is termed "EXPANSION". In contrast to A* algorithm , which follow the path from start to finish , D* begins by searching backwords from the goal node. Each expanded node has a BACKPOINTER which refers to the next node leading to the target and each node knows the exact cost to the target. When the start node is the next node to be expanded , the algorithm is done and the path to the goal can be found by simply ***following the backpointer[10] .***

3.1 D* operator

- The OPEN list is used to propagate information about changes to the arc cost to states in the spaces.
- t(X) tag , every state X has an associated tag t(X) ,D* uses three of tag

t(X)=NEW if X has never been on the OPEN------------------- 1
t(X)=OPEN if X is currently on the OPEN list --------------------2
t(X)= CLOSED if X is no longer on the OPEN list----------------3

- b(X)=Y backpointer (D* uses backpointers to represent paths to the goal) every state X except G has a backpointer to next state Y
- h(G,X) path cost function is an estimate is equivalent to the optimum(minimal) cost from state X to G.
- k(G,X) key function is defined to be the minimum of h(G,X) before modification and all values assumed by h(G,X) since X was placed on the OPEN list. The key function classifies state X on the OPEN list into one of two types a RAISE and LOWER
- K-old is defined to be equal to K-min prior to most recent removal of state from OPEN list . If no have been removed , K-old is undefined.

4- *proposed method*

The proposed algorithm can be divided into three stages , the first two stages described the backward and forward direction , where the third one is the mid-point calculation method.

4.1 Backward direction

In this part the best node computed according to D* algorithm . The algorithm starts with computing of the D* operator for the goal state . The algorithm works by selecting the best node using D* algorithm follows from the goal state and according to its best K-value , the algorithm buts all its neighbors in OPEN-1 list and pop the goal node off and make the new value of K equal to the best h-value of states in OPEN-1 list indicated by the back pointer .

4.2 Forward direction

In this part the best node computed according to A* algorithm . The algorithm starts with computing A* for the start state putting it in the OPEN-2 list . All neighbors of start state are now processed to determine the best one (has minimum value of f(n)) and pop off the start state in CLOSED list .

4.3 Mid-Point calculation

In this part , the mid-point between the best node in OPEN-1 list and the best node in OPEN-2 list according to mid-point formula .The resulted position will be the next node to be searched by changing the value of backpointer to indicate to mid-point , again all neighbors of this state will be evaluated choosing the one with best K-value . This point again will be the best node in A* algorithm so all states generated from this point will be evaluated and the one with minimum f(n) will be selected . This process will be repeated until the back pointer indicates to start states which makes the algorithm stopped successfully .It is important to say that if the node selected according to above method appears to be obstacle the all the points that are affected are again on OPEN-1 list and marked RAISE . Before a RAISED node increase in cost , the algorithm checks its neighbors to determine if its obstacle or not.

Algorithm

1- Start with goal in OPEN-1 list and initial in OPEN-2 list
2- Generate all the neighbors of the node in OPEN-1 list and calculate all D* operators to these points
3- Choose the one with the best h-value indicated by the backpointer
4- Pop the goal form OPEN-1 list and puts in CLOSED.
5- Generate all the neighbors of the node in OPEN-2 list and calculate f(n) to them .
6- Choose the one with the minimum f(n)
7- Nominate the point in the middle distance between the best point in 3 and the best node in 6
8- Check if the nomination node is obstacle RAISE will be alarmed and pop this points off generating all of its neighbors and go to 3
9- Retrieve all the best node in CLOSED list to be the solution path.

5.**Results**

The proposed algorithm applied on 2D work spaces with dimensions described as follows :

Test-1:

The first test applied on 10x10 environment size .The start position is 1,1 where the goal position is 10x10 . There are 34obstacles positions defined by user .The proposed method gives solution path better than others paths resulted by scatter or D* . The path resulted visits positions less than others paths with a 65 percent .

Test-2 :

The second test applied on 15x15 environment size . The start position is 1,1 and the goal position is 15x15. The obstacles were designed as U-shape containing 49 positions . The resulted path is no longer better than other paths , but it does not fall in the trap of U-shape obstacles .

6-Conclusion

From the study and review of existing D* Algorithm, the main problems of the algorithm are mobile robot is traversing across obstacles“ sharp corners, traversing in between two obstacles, and trapped in the U-Shaped type obstacles. Also there is no real time implementation reported. In A* algorithm the mian problem encountered was how to get accurate h(n) since the worst h(n) may lead the algorithm to go to region that its not our interest in the search . To overcome the mentioned problems our method tested in 7x6 2d work space and in some way gives solution paths shorter than the paths resulted from the two algorithms if applied separated . It important to mention that if the work space with larger area the problem of U-shaped can be avoided . The huge computations in larger work spaces will be the main limitation of the proposed method

References

1- Alvarez, Alberto, Caiti, Andrea, & Onken, Reiner. (2004). "Evolutionary Path Planning for Autonomous Underwater Vehicles in a Variable Ocean". IEEE Journal of Oceanic Engineering, Vol. 29, No. 2, Pages 418 – 429.

2- Bräunl, Thomas. (2003). Embedded Robotics: Mobile Robot Design and Applications with Embedded Systems. Springer, Verlag, Berlin, Heidelberg.

3- Choset, Howie, Lynch, Kevin, Hutchinson, Seth, Kantor, George, Burgard, Wolfram, Kavraki, Lydia, & Thrun, Sebastian. (2007). Principles of Robot Motion: Theory, Algorithms, and Implementation.

4- Eustice, Ryan M., Pizarro, Oscar, & Singh, Hanumant. (2008). "Visually Augmented Navigation for Autonomous Underwater Vehicles". IEEE Journal of Oceanic Engineering, Vol. 33, No. 2, Pages 103 – 122. Ferguson, Dave, & Stentz, Anthony. (2006). "The Field D* Algorithm for Improved Path Planning and Replanning in Uniform and Non-uniform Cost Environments". Journal of Field Robotics, Vol. 23, No. 2, Pages 79 – 101.

5- Franz, Matthias O., & Mallot, Hanspeter A.. (2000). "Biomimetic Robot Navigation". Robotics and Autonomous Systems 30, Pages 131 – 153.

6- Fu, Yili, Xu, Hongyan, Li, Han, Wang, Shuguo, & Xu, He. (2006). "A Navigation Strategy based on Global Geographical Planning and Local Feature Positioning for Mobile Robot in Large Unknown Environment". 1st International Symposium on Systems and Control in Aerospace and Astronautics (USSCAA), Pages 1189 – 1193.

7- Garcia, Elena, Jimenez, Maria Antonia, Gonzalez de Santos, Pablo, & Armada, Manuel. (2007). "The Evolution of Robotics Research – From Industrial Robotics to Field and Service Robotics". IEEE Robotics and Automation Magazine, Volume 14, Issue 1, Pages 90 – 103.

8- Hachour, O. (2008). "Path Planning of Autonomous Mobile Robot". International Journal of Systems Applications, Engineering and Development, Issue 4, Volume 2.

9- Howard, Ayanna, Tunstel, Edward, Edwards, Dean, & Carlson, Alan. (2001). "Enhancing Fuzzy Robot Navigation Systems by Mimicking Human Visual Perception of Natural Terrain Traversability". IFSA World Congress and 20th NAFIPS International Conference, Joint 9th, Volume 1, Pages 7 – 12.

10- Koenig, Sven, & Likhachev, Maxim. (2002). "Improved Fast Replanning for Robot Navigation in Unknown Terrain". IEEE International Conference on Robotics and Automation (ICRA '02) , Volume 1, Pages 968 – 75.