Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 23 (2016) 144 – 149

Procedia Technology

3rd International Conference on Innovations in Automation and Mechatronics Engineering, ICIAME 2016

Time-Efficient A* Algorithm for Robot Path Planning

Akshay Kumar Guruji, Himansh Agarwal, D. K. Parsediya*

Madhav Institute of Science & Technology, Race course road, Gwalior, Madhya Pradesh, India

Abstract

The current era is mainly focused on the modernization, industrialization, automation and development. For which, the human task are replaced by robots to achieve good accuracy, high efficiency, speed and multiplicity. In industries, these robots are employed to carry heavy objects in working place. As the environment or working area may be dynamically changing, the algorithm or the rules must be devised to ensure an optimistic collision-free path. A* algorithm is a heuristic function based algorithm for proper path planning. It calculates heuristic function's value at each node on the work area and involves the checking of too many adjacent nodes for finding the optimal solution with zero probability of collision. Hence, it takes much processing time and decreases the work speed. The modifications in A* algorithm for reducing the processing time are proposed in this paper. The proposed A* algorithm determines the heuristic function's value just before the collision phase rather than initially and exhibits a good decrement in processing time with higher speed. This paper involves MATLAB simulation of robot movement from source to goal. Several cases are considered with proposed A* algorithm which exhibit maximum 95% reduction in processing time.

Peer-review under responsibility of the organizing committee of ICIAME 2016

Keywords: Path planning, optimum path, heuristic function, processing time and path length.

1. Introduction

Path planning is the method of finding the optimal path from one point to another in space. It optimizes the path between source and destination by determining shortest path between them [1-2]. It is sometimes also termed as

* Corresponding author. Tel.:+91 9827579912.
E-mail address: parsediyadeep@gmail.com

Peer-review under responsibility of the organizing committee of ICIAME 2016
doi:10.1016/j.protcy.2016.03.010

motion planning as it helps to decide the motion of any object in the working environment. An object can be a robot which is autonomous in nature as it makes use of the path finding algorithm to determine it traversing points in space. Such a robot is referred to as mobile robot [3]. Path planning can also be defined as the process of breaking down a desired path movement into number of iterative steps to make discrete motions to optimize some entities.

Environment plays an important role in path planning problems. Based on nature of the workplace, path planning can be classified as offline and online. In former, the data regarding workspace containing stationary obstacles of which the geometry is known which are given as input and there after the path is found using the algorithm [4]. While in later, the robot uses sensors or real time data acquiring equipments in order to find the position of the obstacle that constantly move in the entire workplace. A lot of work from different authors and publications suggest this [5-8]. But implementation of such concept requires sophisticated modelling. In this paper the conditions deal with the static environment therefore the devised algorithm is offline in nature. Various path searching algorithms such as Genetic, ANN and A* algorithm are employed for mobile robot navigation as in [9]. It assumes the complete workplace as a grid. The robot determines the path that covers least number of grid cells to reach destination. A* algorithm makes use of the heuristic function and calculates its value at each node on the work area to get the optimal solution [10]. A* algorithm also suffers from certain limitations. Therefore prior to this study various variants of the A* algorithm have been devised [11-13].

A* is a computer algorithm. It must need processors of high configuration in order to check various nodes successively. If the distance between the source and the goal is more, then implementation of A* algorithm from the source and the goal involves the checking of too many adjacent nodes one after another as in [14] thus increasing the processing time. In this case processors with fewer configurations require excess of time to find the path. The above problem has been solved by following a new algorithm which is proposed in this paper. This research work designed a computer algorithm that is a variant of A* algorithm and which reduces the processing time. The proposed algorithm does not determine the heuristic function's values for each node. It measures the value only just before collision phase and process frequently. In case where old processors are available this algorithm can also be employed. The main focus is also not to lose the beneficial aspects of conventional A* algorithm. If the distance between the source and the goal is more this new modified algorithm named as time efficient A* algorithm can be employed which offers a huge reduction such as 75% in the execution time.

The paper consists of various sections. Section II describes the proposed algorithm along with conditions. Section III illustrates the simulations and results obtained with comparative study. Sections IV consists the conclusion.

2. Proposed algorithm and conditions

The conventional algorithm takes much time for fetching all nodes and for calculations of heuristics function's values. The proposed Time Efficient A* algorithm fetches all nodes but calculates the heuristics function's value only before collision phase. This reduces the processing time so that the robot can perform its work quickly. Figure 1 illustrates the flow chart of the proposed Time Efficient A* algorithm. The proposed methodology is divided in 5 phases of operation. This methodology is followed in MATLAB simulation for estimating the processing time.

2.1. Input phase

At first the image of the robot's work arena is used and converted to binary image where the black region shows the obstacles. The image is considered to be placed symmetrically at the origin of x & y-plane. Give the coordinates of starting point (source) and the destination (goal) as inputs.

2.2. Proximity check phase

This involves the checking of the distance between the source and goal. If the distance is less it means the proximity condition is satisfied then proceed to the A* phase otherwise proceed as further.

2.3. Slope determination and pixel projection phase

Slope of the line containing the source and goal as its ends point is calculated. If slope calculated is unity, then the pixels are projected on the image from the source on the imaginary line of connection between the source and the goal. The condition using the coordinates of source and goal is checked whether the modified A* algorithm is useful or not for the set of those coordinates. If source and goal lie close to each other then original A* is applied as it is beneficial but in case the distance is more along with a obstacle in between the modified A* algorithm is better. If the slope is not unity then pixel is projected at an arbitrary slope calculated by values of h and g.

2.4. Collision phase

If the pixels projection reaches the goal the straight line path is obtained between the source and the goal. If the projection strikes the obstacle then the coordinates of the point just before the collision are obtained it is said to be switching point (abbreviated as S point) as after S point the algorithm shifts to A* algorithm.

2.5. A or Switching phase*

Using point S as source and the destination as goal the A* is applied to obtained the shortest path. This is named as switching phase because the algorithm now starts to predetermined A* algorithm for path planning. The combination of the straight line path from source to point S and A* algorithm resultant path from Point S to goal gives the total path. Path obtained from step 10 is then fed to the robot to move from source to destination.

The above mentioned approach gives optimum result when certain conditions are met. The work place is considered as static. Geometry and position of the robot are known. The coordinates or relative positions of source, goal and robot are known prior to the application of the proposed algorithm.

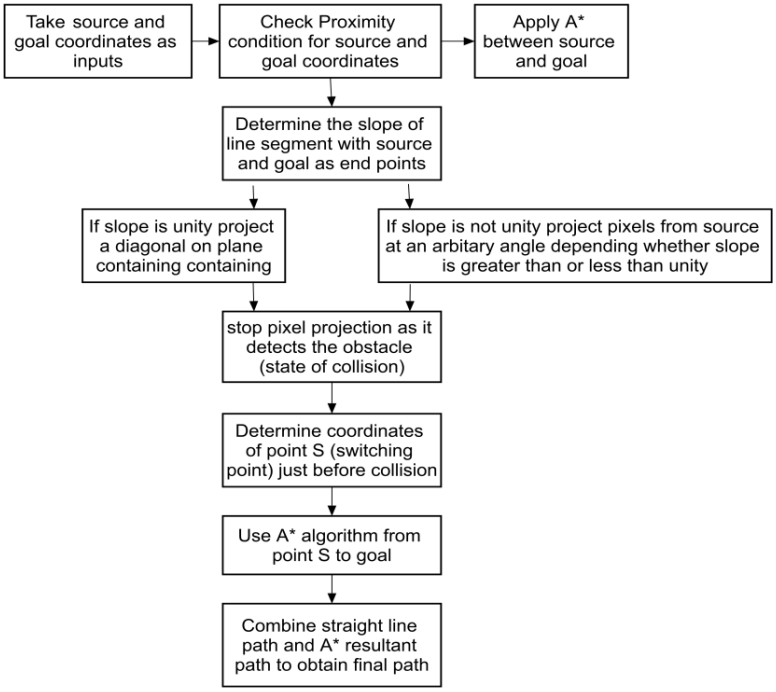

Figure 1: Flow chart of proposed time-efficient A* algorithm

3. Simulation and Results

The proposed algorithm is implemented on MATLAB version 7.12.0.635(R2011a). In which various graphics functions are employed. The image of arena taken by overhead camera (as discussed in previous section) is converted into binary form or bitmap image. The processing time and path length are obtained as output from the MATLAB program. The complete code is then tested on Intel Pentium CPU B9600 @ 2.20GHz indicating the algorithm's reliability on slow processors.

Table 1: Cases and Parameters

S. No.	Case	Initial position of robot (coordinates of source)	Final position of robot (Coordinates of destination)
1	1	Source=[5,5]	Goal=[90,90]
2	2	Source=[5,5]	Goal=[70,20]
3	3	Source=[5,5]	Goal=[40,80]
4	4	Source=[5,5]	Goal=[70,70]
5	5	Source=[5,5]	Goal=[35,90]
6	6	Source=[5,5]	Goal=[50,50]

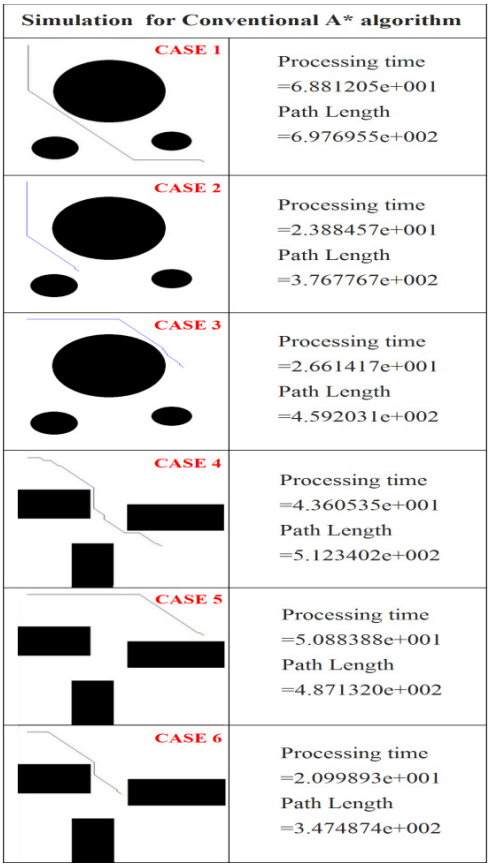

Figure 2: Simulation for conventional A* algorithm

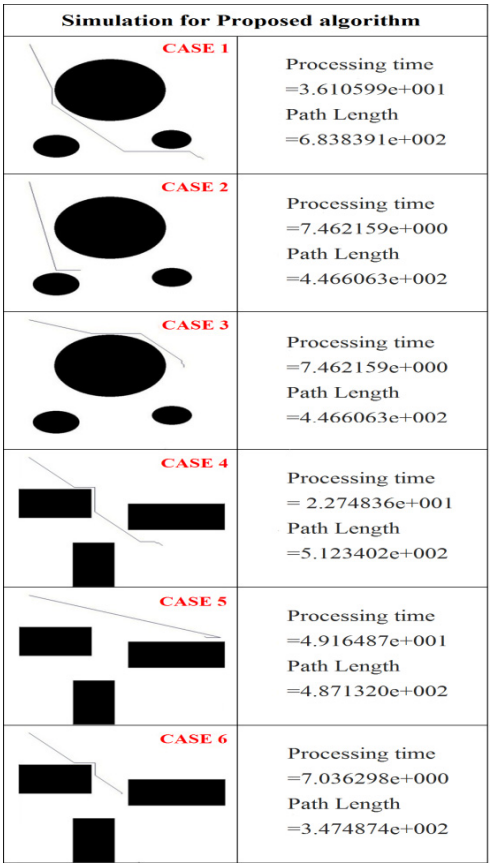

Figure 3: Simulation for the proposed time-efficient A* algorithm

Table 1 illustrates the different cases used for simulation. The robots starts its traversing from the initial position named as source and approaches destination known as goal. The source positions are fixed for all the cases. The table enlists the coordinates of source and goal.

Figure 2 illustrates the results of simulation for conventional A* algorithm using parameters enlisted in table 1. The simulation results for the proposed algorithm are depicted in Figure 3 for the same set of parameters. Both figures show the traversing paths from source to goal along with processing time and path length written next to it. The study comprises the simulation for several cases which concludes the time efficiency of proposed algorithm.

Table 2: Simulation results and Comparison

S. No	Conventional A* algorithm		Proposed algorithm		Comparison	
	Processing time	Path length	Processing time	Path length	% decrease in processing time	% increase in path length
1	6.881205e+001	6.976955e+002	3.610599e+001	6.838391e+002	47.54	1.98
2	2.388457e+001	3.767767e+002	1.280379e+000	3.914363e+002	94.6	3.89
3	2.661417e+001	4.592031e+002	7.462159e+000	4.466063e+002	71.96	2.74
4	4.360535e+001	5.123402e+002	2.274836e+001	5.123402e+002	47.83	0
5	5.088388e+001	4.871320e+002	4.916487e+001	4.871320e+002	3.38	0
6	2.099893e+001	3.474874e+002	7.036298e+000	3.474874e+002	66.43	0

The observations from simulations are listed in Table 2. The results of the proposed algorithm are then compared with the conventional algorithm and the percentage change is noted for processing time and path length. The observations are as follows;

- For most of the cases the remarkable reduction in processing time is noted for small increase in path length. It is beneficial while dealing with slow processors. The proposed algorithm takes less than half the time taken by the conventional A* algorithm which compensates the mere increase in path length.
- In some cases the both the processing time and path length are found less with the proposed algorithm thereby providing more optimum path than the conventional algorithm.
- Sometimes the processing time calculated for proposed algorithm is found less than the conventional algorithm for the same path length therefore it is a optimum solution

4. Conclusion

The MATLAB programming of conventional A* algorithm and proposed time-efficient A* algorithm has been done in this research work which exhibits comparison between path searching methods for both the algorithms. The percentage increment in path length and percentage decrement in processing time are observed and calculated for both the algorithm. In this paper, it is obtained that modified A* algorithm reduces the processing time at least by 65% with utmost 3.4% increase in path length. Modified A*algorithm is better than original A* algorithm when the distance between the source and the first obstacle is more than the straight line path and reduces checking of each adjacent values being used in heuristic employed algorithm like A* algorithm. Thereby it can be concluded that the modified A* algorithm is better than the actual A* algorithm in terms of processing time on a little cost of path length and can be applicable for fast processing applications.

References

[1] S. A. Hutchinson, R. L. Cromwell, and A. C. Kak: Planning Sensing Strategies in a Robot York Cell with Multi-Sensor Capabilities IEEE Trans on Robotics and Automation 1988 IEEE.

[2] Alexopoulos, C.: path planning for mobile robots Systems, Man and Cybernetics, IEEE Transactions on (Volume:22 , Issue: 2)

[3] Janusz Pochmara, Wojciech Grygiel, Radosław Koppa, Krzysztof Kamiński "Mobile Robot Platform for Real-Time Search Algorithms" ; International Conference "Mixed Design of Integrated Circuits and Systems" , June 20-22, 2013, Gdynia, Poland

[4] Ismail, AL-Taharwa, Alaa Sheta, and Mohammed Al-Weshah. "A mobile robot path planning using genetic algorithm in static environment." *Journal of Computer Science* 4.4 (2008): 341-344.

[5] Seungho Lee' and Teresa M. Adamsb : A PATH PLANNING ALGORITHM FOR AUTOMATED CONSTRUCTION EQUIPMENT ,automation and Robotics in Construction XVI 1999 by .CC3M.

[6] Beom, H.R. , Koh, K.C. ; Cho, H.S.: Behavioral control in mobile robot navigation using fuzzy decision making approach Intelligent Robots and Systems '94.

[7] Yasuda, G.; Takai, H., "Sensor-based path planning and intelligent steering control of nonholonomic mobile robots," *Industrial Electronics Society, 2001. IECON '01. The 27th Annual Conference of the IEEE* ,

[8] Woong-Gie Han; Seung-Min Baek; Tae-Yong Kuc, "GA based online path planning of mobile robots playing soccer games," *Circuits and Systems, 1997. Proceedings of the 40th Midwest Symposium on* , vol.1,

[9] R. Kala, A. Shukla, R. Tiwari, S. Roongta, R. R. Janghel (2009) Mobile Robot Navigation Control in Moving Obstacle Environment using Genetic Algorithm, Artificial Neural Networks and A* Algorithm, Proceedings of the IEEE World Congress on Computer Science and Information Engineering, Los Angeles/Anaheim, USA, pp 705-713.

[10] DaveFerguson, MaximLikhachev, and Anthony Stentz "A Guide to Heuristic based Path Planning"

[11] WANG Zhu, LIU Li, LONG Teng, YU Chenglong, KOU Jiaxun "Enhanced Sparse A* Search for UAV Path Planning Using Dubins Path Estimation"; Proceedings of the 33rd Chinese Control Conference July 28-30, 2014, Nanjing, China

[12] Liping Cheng and Chuanxi Liu, Bo Yan "Improved Hierarchical A-star Algorithm for Optimal Parking Path Planning of the Large Parking Lot"; Proceeding of the IEEE International Conference on Information and Automation Hailar, China, July 2014

[13] Elisabete Fernandes, Pedro Costa, Jos´e Lima, Germano Veiga "Towards an Orientation Enhanced Astar Algorithm for Robotic Navigation"; 2015 IEEE

[14] Zhou Weiteng, Han Baoming , Li Dewei, Zheng Bin "Improved Reversely A star Path Search Algorithm based on the Comparison in Valuation of Shared Neighbor Nodes"; 2013 Fourth International Conference on Intelligent Control and Information Processing (ICICIP) June 9 – 11, 2013, Beijing, China