Proc. of the Second Intl. Conf. on Advances In Computing, Electronics and Communication - ACEC 2014.
Copyright © Institute of Research Engineers and Doctors, USA .All rights reserved.
ISBN: 978-1-63248-029-3 doi: 10.15224/ 978-1-63248-029-3-72

P*: A New Path Planning Algorithm for Autonomous Robot in an Unknown Environment

Farouk MEDDAH, Lynda DIB*

***Abstract*— In this paper we present P* "P-Star" witch is a new algorithm for sensor based path planning.**

This work is based on PointBug algorithm [11] where we applied some improvements and modifications to overcome some important problems (like infinite loops and the bypass of some sub-paths).

Moreover, we present some simulation results and comparisons with PointBug to evaluate and to verify the performance and the power of the proposed algorithm.

Keywords*— *Mobile Robot, Path Planning, Unknown Environment, PointBug.

I. Introduction

Path planning in an unknown environment is one of the most challenging problems in robotics. A lot of algorithms were proposed for both finding and optimizing the path between two points "source" and "target".

Bug algorithms are based on two behaviors: move toward the target and avoid encountered obstacles. The difference between these algorithms is in leaving conditions [3] [4] [5].

In Bug2 algorithm [1] [2] the robot moves toward the target, when encountering an obstacle the robot follows the boundary of this obstacle, until meeting the line that crosses the start and the target point. After, the robot continues moving toward the target.

Bug2 algorithm weakness was improved by Alg1 [6] and Alg2 [7] [13], which records hit and leave points to avoid tracing the same path twice, with a little differences.

In TangentBug algorithm [8] [9] [10], the robot moves in a straight line toward the target, until encountering an obstacle the robot starts the boundary-following behavior. To do so, TangentBug uses local tangent graph. The robot stops moving along the boundary of the obstacle once it founds a point in tangent graph closer to target than the boundary's point. At that point, the robot continues its path toward the target.

Dynamic Point Bug algorithm [15] uses another strategy. the robot advances and adjusts its direction continuously to the direction of the target, and when the robot encounters an obstacle it moves to right or left to avoid this obstacle, then it continues its path toward the target.

Another powerful algorithm is PointBug algorithm. Its principle strategy is few different of the others when it introduces the notion of sudden points. As defined in [11], *sudden point* is a point where a sudden change in distance of sensor's range is detected, there is three possible changes: (a) from infinite to some distance in some sides of obstacles, (b) from some distance to infinite distance in other sides of obstacles, (c) from some distance to some distance when an obstacle hides a part of itself or of another obstacle. The first direction of robot is facing the target point where it starts searching for the first sudden point. After, the robot repeats these two actions: (1) Moves towards next sudden point; (2) rotates in the direction of *dmin-line* (the line that crosses the current and the target points) for searching the next sudden point. The robot stops its displacement once it reaches the target or when no sudden points were found. When the robot searches sudden points it ignores sensor reading at rotation of 180° to avoid detecting previous sudden points.

PointBug algorithm is reliable for an environment where there are not many obstacles and no spiral (it may contain one or more simple minimum points), some others reachability limitation are presented in [12] [14].

PointBug algorithm assumes using infinite sensor range witch does not exist in real world.

This paper is organized such as follows: As P* is based on PointBug algorithm, so, in section 2 a set of PointBug's limitations will be presented. In section 3 we describe our new algorithm for sensor based path planning. Finally, a set of experiments will be presented and discussed.

II. PointBug algorithm limitations

PointBug algorithm has several limitations. In this paper we try to overcome its most important problems when we improve this algorithm by applying some modifications and using some new definitions.

As first problem, there are no tests in PointBug algorithm if a sudden point was already treated (since it does not record visited sudden points). This situation can produce infinite loops in case where the target is surrounded by uniform boundary obstacle, see Figure 1.

Farouk MEDDAH *(Author)*

LASE: Laboratoire des Systèmes Embarqués. University of Badji Mokhtar, BP 12, 23000 Annaba, Algeria.

Lynda DIB *(Author)*

LASE: Laboratoire des Systèmes Embarqués. University of Badji Mokhtar, BP 12, 23000 Annaba, Algeria.

SEEK DIGITAL LIBRARY

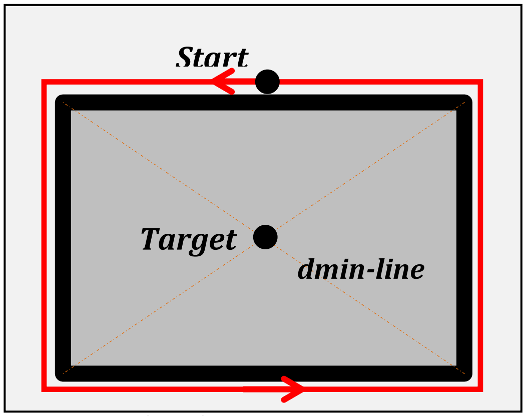

Figure 1: A case of an infinite loop produced by PointBug algorithm.

As second problem of this algorithm is the use of unlimited sensor range. The robot always detects the most far sudden points at first. If there are one or more sudden points closer to the current point they will be detected after. In this case, the robot can bypass some paths that may lead to the target or even miss the alone way to destination, see Figure 2.

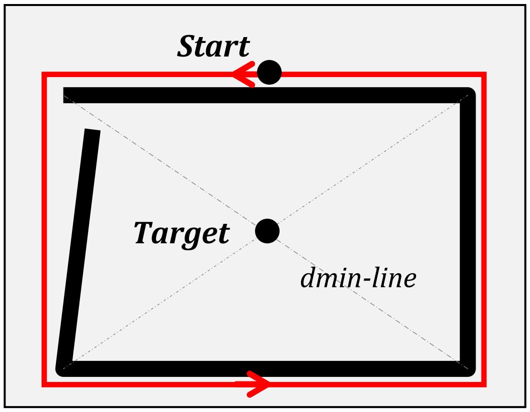

Figure 2: example where PointBug algorithm can not reach the target.

III. Proposed Algorithm P*

As stated previously, a lot of algorithms were proposed for both finding and optimizing a path between two points “source and target”. In this work, we propose a new algorithm called “*P* : PStar algorithm*” which is based on PointBug but by providing it some modifications that ensures the resolution of its major limitations.

A. *Solution of endless cycle problem*

In the first part, we propose a solution to the endless cycles and double treatments of sudden points. For this, the condition of 180° bound is removed; the points above 180° will not be ignored. Also, we will record the sudden points and just ignore the sudden points already treated and their sons as well (all their succeeding sudden points). The algorithm follows the same strategy with all points. So, the path generated will be the same as the previous one. As a result the current path will be ignored completely. When the robot finds a treated sudden point once again, it ignores this point and continues the treatment of the succeeding sudden point. If there is no sudden point, the robot return to previous one, until reaching the target or no more sudden point can be found from the first one.

B. *Description of P* algorithm*

```
Begin
    (1) FindNextDirection and advance to selected direction,
        until one of these events occurs:
    (2) The robot reaches the target Stop.
    (3) The first sudden point was encountered again and all
        sudden points were treated: Stop, the target is not
        reachable.
    (4) The robot must respect these rules:
        • If no sudden point was found yet treat the first hit
          point as a sudden point.
        • The adjustment of the direction angle must not
          exceed defined bound value when following
          obstacle (this bound is a parameter).
        • Don't turn back until it's impossible to advance.
        • The current sudden point was already treated
          ignore all its sons (next sudden points detected
          from it).
    (5) If a new sudden point was found, go to (1)
End
FindNextDirection
    If No sudden point can found Then /* due to limited sensor
range */
    The robot tries to move directly as possible towards the
target along dmin-line, if the robot starts travelling away of
target it follows the boundary of the current obstacle
    Else
    From the current point, the robot numerates all sudden
points can found and goes to the first one.
    Endif
    If the selected sudden point already visited ignore it and go
to the next one.
End
```

IV. Simulation and discussion

In this section we simulate our proposed P* algorithm. The generated robot’s trajectories will be compared with PointBug simulations results.

A. *Simulation comparison when no problem of PointBug exists*

Simulation 1: General case

Using limited sensor range gives different results. Moreover, the detected sudden points may differ. The following simulation shows the difference between the results obtained by PointBug algorithm (with unlimited range sensor –as default-) and P* algorithm (with limited range sensor).

Proc. of the Second Intl. Conf. on Advances In Computing, Electronics and Communication - ACEC 2014.
Copyright © Institute of Research Engineers and Doctors, USA .All rights reserved.
ISBN: 978-1-63248-029-3 doi: 10.15224/ 978-1-63248-029-3-72

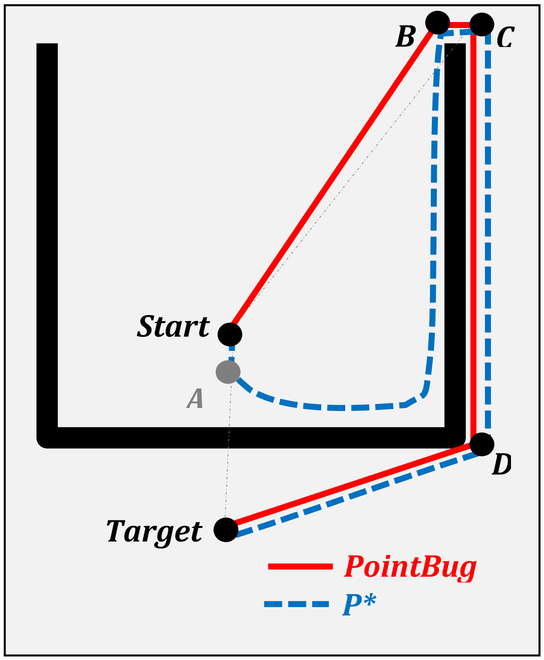

Figure 3: Trajectory generated using PointBug and P* algorithm in a simple Office like Environment (Environment and PointBug trajectory are taken form [3]).

In the case simulated in Figure 3 there are no problems of PointBug algorithm cited in section 2.

In this simulation, we see that the length of the trajectory generated by P* algorithm (dotted curve) converges to the trajectory generated by PointBug algorithm more and more with the augmentation of robot's sensor range.

Simulation 2: When non-straight boundary obstacles encountered

In this case, PointBug algorithm will detect at least one sudden Point, but P* algorithm may not detect any sudden point and the generated path will be better.

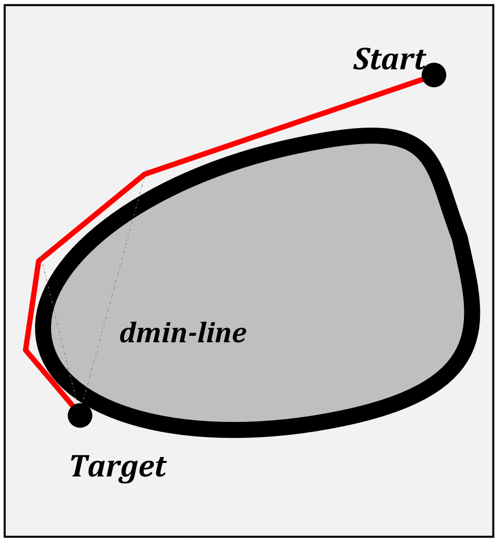

Figure 4: PointBug follows a non-straight obstacle boundary.

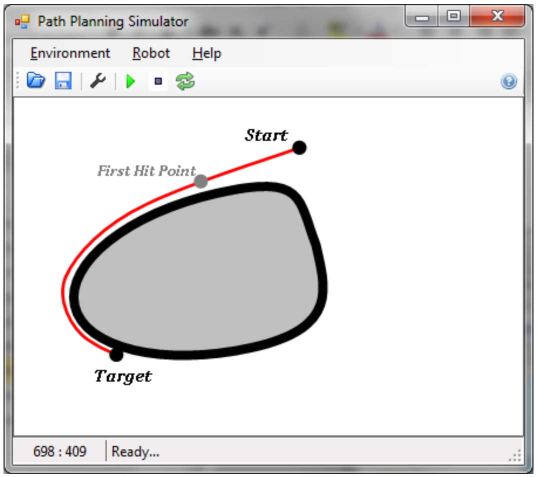

Figure 5: P* follows the previous non-straight obstacle boundary.

Figure 4 shows the trajectory of robot resulting by simulating PointBug algorithm and where three sudden points were detected. Nevertheless, Figure 5 shows the trajectory of the robot when P* algorithm is simulated. In this case we see there are no sudden points and the trajectory is better than that in PointBug algorithm.

The above simulation of Pstar algorithm gives 1138mm vs. 1305mm PointBug's trajectory length.

B. *Simulation when there is a problem of PointBug*

Simulation of bypassing some sub-paths by PointBug

The use of limited range sensor does not necessarily give worse results. Simulation generated in Figure 6 and Figure 7 show a case where the limited range sensor gives a good result while using unlimited range sensor doesn't give a result at all.

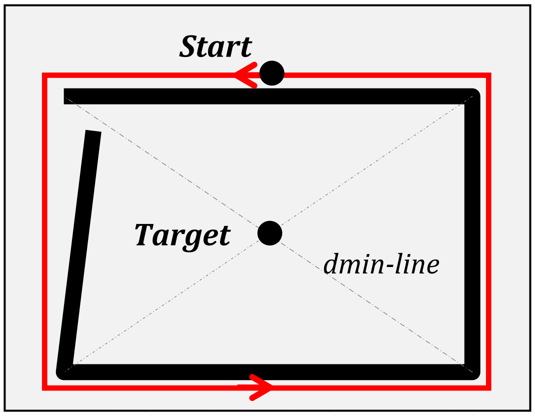

Figure 6: Example where PointBug algorithm can't reach the target.

SEEK DIGITAL LIBRARY

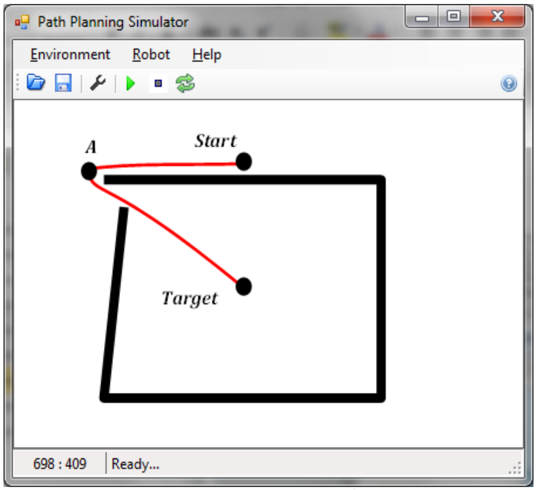

Figure 7: P* algorithm gives a solution for the previous problem.

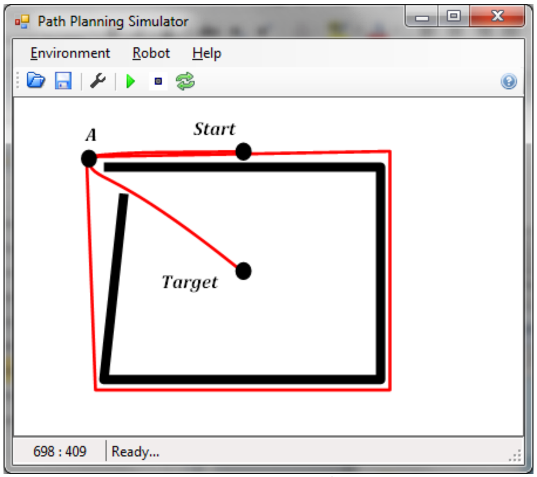

Figure 8 P* algorithm gives a solution for the same problem (**Error! Reference source not found.**) using a limited range sensor (greater distance).

Figure 6 shows the simulation of PointBug algorithm. In this simulation, when the robot reaches the first sudden point, it will detect the one in the opposite corner first. So, instead of going directly toward the target, the robot goes to the opposite corner and continues turning infinitely around it.

Figure 7 shows the simulation of P* algorithm. In this simulation we used a limited range sensor. As result, the next corner will not be detected and the robot goes directly toward target.

In the worst case (example Figure 8), P* algorithm will generate one turn only, because when the robot arrives at the second time to the first sudden point it ignores the next sudden point with its sons and goes toward the target. This case occurs when the ranger sensor is greater than distance between the robot and the most far sudden point.

V. Conclusion

In this paper we presented a new algorithm P*, which can overcome the problems of PointBug algorithm. In the general case and in the absence of the problems of the PointBug algorithm, by increasing the sensor's range the trajectory generated by PointBug algorithm converges to the trajectory generated by P* algorithm. This difference is due to the use of unlimited range sensor by PointBug algorithm.

Another feature of P* algorithm is it can be used directly in a maze without any problem and without changes.

In future, we hope to present an experiment on a real physical environment.

References

[1] V. Lumelsky and P. Stepanov: "Dynamic Path Planning for a Mobile Automaton with Limited Information on the Environment". IEEE Transactions on Automatic Control Volume 31 No. 11. Pages 1058-1063. Nov 1986.

[2] Vladimir J. Lumelsky and Alexander A. Stepanov: "Path-Planning Stategies for a Point Mobile Automaton Moving Amidst Unknown Obstacles of Arbitrary Shape". Algorithmica (1987) 2: pages 403-430.

[3] Vladimir J. Lumelsky: "A Comparative Study on the Path Length Performance of Maze-Searching and Robot Motion Planning Algorithms". IEEE Transactions on Robotics and Automation, Vol. I, No. 1, Feb. 1991. Pages 57-66.

[4] Alpaslan Yufka et Osman Parlaktuna : "Performance Comparison of Bug Algorithms for Mobile Robots". 5th International Advanced Technologies Symposium (IATS'09), May 13-15, 2009, Karabuk, Turkey.

[5] Liam Paull: "Robust Online Adaptive Sensor-Driven Survey Planning for Single and Multiple Autonomous Underwater Vehicles". PhD Thesis. University of New Brunswick. Octobre 2013.

[6] A. Sankaranarayanan et M. Vidyasagar. "A New Path Planning Algorithm for Moving a Point Object Amidst Unknown Obstacles In A Plane". IEEE Conference on Robotics and Automation, Pages 1930-1936, 1990

[7] A. Sankaranarayanan and M. Vidyasagar: "Path Planning for Moving a Point Object Amidst Unknown Obstacles In A Plane: The Universal Lower Bound On Worst Case Path Lenghts And A Classification of Algorithms". IEEE International Conference on Robotics and Automation, Sacramento, California, April 1991. Pages 1734-1741.

[8] K. Ishay, R. Elon and R. Ehud: "TangentBug: A Range-Sensor-Based Navigation Algorithm". The International Journal of Robotics Research. Pages 934-953. 1998.

[9] K. Ishay and R. Elon: "A New Range-Sensor Based Globally Convergent Navigation Algorithm for Mobile Robots". Techion Department of Computer Science. 1995.

[10] James Ng. "An Analysis of Mobile Robot Navigation Algorithms in Unknown Environments". Ph.D. thesis, School of Electrical, Electronic and Computer Engineering. Feb. 2010.

[11] Buniyamin N., Wan Ngah W.A.J., Sariff N., Mohamad Z: "A Simple Local Path Planning Algorithm for Autonomous Mobile Robots". International Journal of Systems Applications Engineering and Development. Issue 2, Volume 5, 2011. Pages 151-159.

[12] Noborio, H. and Yoshioka, T.: "An on-line and deadlock-free path-planning algorithm based on world topology". Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems 93, IROS '93. Yokohama, Japan. July 26-30, 1993. Pages 1425-1430.

[13] A. Sankaranarayanan et M. Vidyasagar: "Path Planning for Moving a Point Object Amidst Unknown Obstacles In A Plane: A New Algorithm And A General Theory For Algorithm Development". IEEE 29th International Conference on Decision and Control, Honolulu, Hawaii, December 1990. Pages 1111-1119.

[14] Michael Colin Hoy: "Methods for Collision-Free Navigation of Multiple Mobile Robots in Unknown Cluttered Environments". Ph.D thesis. School of Electrical Engineering and Telecommunications, University of New South Wales. January 2013.

[15] B. Margaret Devi, Prabakar S, “Dynamic Point Bug Algorithm For Robot Navigation”, International Journal of Scientific & Engineering Research, Volume 4, Issue 4, April-2013, pages 1276-1279.

SEEK
DIGITAL LIBRARY