AIP Conference Proceedings

Coverage path planning optimization based on Q-learning algorithm

Cite as: AIP Conference Proceedings **2116**, 220002 (2019); https://doi.org/10.1063/1.5114220
Published Online: 24 July 2019

Luis Piardi, José Lima, Ana I. Pereira, and Paulo Costa

View Online Export Citation

ARTICLES YOU MAY BE INTERESTED IN

The use of the cosine function in modelling of a ski jumping in-run hill
AIP Conference Proceedings **2116**, 250006 (2019); https://doi.org/10.1063/1.5114246

Mechanical defects in solar cells can be found using advanced image processing
Scilight **2019**, 300006 (2019); https://doi.org/10.1063/1.5120513

Experimental investigations on thermal management and performance improvement of solar PV panel using a phase change material
AIP Conference Proceedings **2128**, 020023 (2019); https://doi.org/10.1063/1.5117935

AIP Publishing

© 2019 Author(s).

Coverage Path Planning Optimization Based on Q-Learning Algorithm

Luis Piardi[1,a)], José Lima[1,2,b)], Ana I. Pereira[1,3,c)] and Paulo Costa[2,4,d)]

[1]*Research Centre in Digitalization and Intelligent Robotics (CeDRI), IPB, Bragança, Portugal*
[2]*INESC-TEC, Centre for Robotics in Industry and Intelligent Systems, Porto, Portugal*
[3] *Algoritmi R&D Centre, University of Minho, Braga, Portugal*
[4] *Faculty of Engineering of University of Porto, Porto, Portugal*

[a)]piardi@ipb.pt
[b)]jllima@ipb.pt
[c)]apereira@ipb.pt
[d)]paco@fe.up.pt

Abstract. Mobile robot applications are increasing its usability in industries and services (examples: vacuum cleaning, painting and farming robots, among others). Some of the applications require that the robot moves in an environment between two positions while others require that the robot scans all the positions (Coverage Path Planning). Optimizing the traveled distance or the time to scan the path, should be done in order to reduce the costs. This paper addresses an optimization approach of the coverage path planning using Q-Learning algorithm. Comparison with other methods allows to validate the methodology.

INTRODUCTION

Path planning is one of the most important and challenging areas of mobile robotics. It is evident that for a mobile robot to successfully perform the task for which it was developed, a robust, efficient and fast solution route planning strategy is essential. Typical path planning problems involve locating the smallest route between two distinct points (start and goal). However, many robotic applications require a path that passes through all points of an environment, such as vacuum cleaning robot, painter robot, demining robots, automated lawn mowers, farming robots, among others [5, 7, 6]. For this reason, a strategy is required to perform a Coverage Path Planning (CPP) and this problem will be addressed in this work using Q-learning algorithm.

The main objective of the Coverage Path Planning is to find a path that passes in all points of an interest area, avoiding obstacles [5], and optimizing the traveled distance or the time needed. CPP is addressed in the literature to be NP-hard problem [1] needing complex algorithms to solve the problem.

In this work, the interest region that the robot will scan is represented by a grid, since it is the approach most adopted by researchers in the area due to its simplicity to represent internal environments [1]. In [11] a pioneering grid-based method to solve the CPP is presented, where the calculation of the distance transform propagating the wavefront from a goal to the start point establishing a value for each cell of the grid. As there are many cells with equal values, the robot ends up visiting the same cell more than once to perform the environment coverage. Also, in the recent work [4], it is used an approach similar to previous, however the values of the cells are updated as the robot progresses within the map. It is a fast convergence approach, however due to similar values in neighboring cells it occurs that the robot visits more than once the same grid cell. An approach using evolutionary algorithms is presented in [9], where the objective is to create a spatial mapping of toxic substances in an internal environment. The resulting pathway is suboptimal due to the initial population of the genetic algorithm be random. An approach to CPP using the well-known A* algorithm (finds a smaller path between two points given a cost function) is described in [3].

The main contribution of this work is to demonstrate a new approach to the reinforcement learning algorithm, named Q-learning, to perform the CPP of an environment with known obstacles, visiting only once each cell of the grid, resulting in an optimized path.

International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2018)
AIP Conf. Proc. 2116, 220002-1–220002-4; https://doi.org/10.1063/1.5114220
Published by AIP Publishing. 978-0-7354-1854-7/$30.00

The paper is organized as follows. The first section presents the CPP concept and its applications. The second section will present the approach used with the Q-learning algorithm to perform the CPP and the obtained numerical results are presented in the third section. The last section shows some conclusions and some ideas for future work.

Q-LEARNING ALGORITHM APPLICATION

To solve the CPP, a representation of the environment and the known obstacles are decomposed into a collection of uniform square grid cells (see Figure 1.(a)), where the size of each cell must be adjusted according to the size of the mobile robot adopted. The starting cell of the path is always defined by the user, and the last visited cell in the grid is dynamic, meaning that the algorithm stops when all cells are visited. The robot does not perform movement in diagonal directions. So, it only moves between interconnected cells in horizontal and vertical directions.

The CPP solution, proposed in this work, is based on Reinforcement Learning (RL), a class of Machine Learning that rests on the principle of reward and punishment [8]. The used RL technique is Q-learning algorithm. More details about Q-learning can be found in [2, 10]. For the solution of CPP, the classical model of updating of the Q matrix was applied, using a random exploration.

The classical model of Q-learning uses a bi-dimensional reward matrix R as reference representing the possible connections between cells (see Figure 1.(b)). The rows are the states S and the columns the actions a, in another words, the rows represent the current cells and the columns indicate cells that connect to the current cell. The dimension of R is $n \times n$ where n express the amount of cells needed to represent the environment. Therefore, for a given environment, there are $S_1, S_2, ..., S_n$ possible states where each state has a maximum of 4 actions (up, down, left and right). A Q matrix, of the same size of R, is used to calculate and memorize the quality of combinations between states and actions during the exploration. All Q positions are initially *Null*. The Q matrix will acquire the experience according to the state transition policy described by Expression 1 and the *repeat* structure in Algorithm 1. The constant γ represents the learning factor, S' represents the next state due to action a and finally a' represents all actions of the next state S'.

$$Q(S, a) = R(S, a) + \gamma * max(Q(S', a')) \tag{1}$$

Thus, the value of $Q(S, a)$ results in the sum of the reward of $R(S, a)$, and the maximum reward for the future states. When the number of episodes for updating Q is reached the Q matrix will store numeric information about the environment from which the solution (path) to the CPP will be extracted (see Figure 1.(c)).

The CPP solution is obtained through the following procedure. Considering the start cell, select the action a_j with the lowest value referring to the state S_i to get the next state and then erase, from the Q matrix, the line for the state S_i and the column for action a_i. Continue the process until terminate all the states. The sequence of the selected states S will identify the sequence of cells visited by the robot to perform the CPP (see Figure 1.(d)). Figure 1 provides an example of a reduced environment represented by 9-cell in 3×3 connectivity grid.

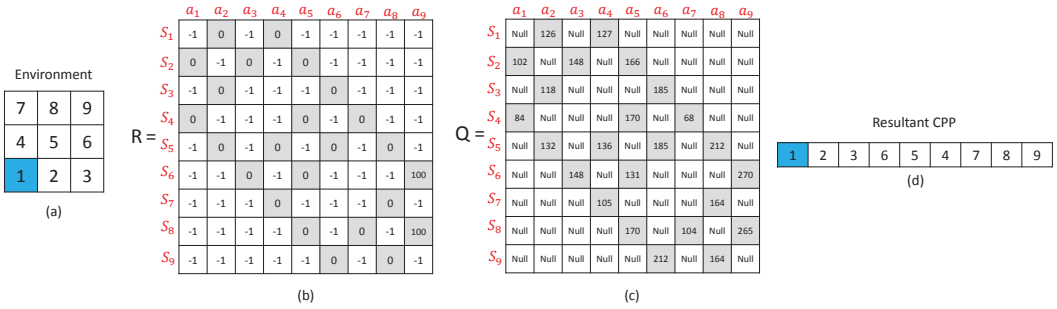

FIGURE 1. CPP for a small environment : (a) Connectivity grid. (b) Reward matrix. (c) Quality matrix after training. (d) Resultant path.

NUMERICAL RESULTS

In this section, two environments (Figure 2) to perform the CPP are presented and solved with the proposed Q-learning method. The numerical results are compared with other strategies. Depending on the algorithms, some positions are

Algorithm 1: Q-learning to solve the CPP

Initialize $R(S, a)$ for all S, a;
Initialize $Q(S, a) = Null \ \forall \ S, a$;
repeat
  S = random S ;
  repeat
    a = random a valid in the current state S ;
    $S' = a$;
    $Q(S, a) = R(S, a) + \gamma * max(Q(S', a'))$;
    $S = S'$;
  until *find a final S*;
until *reach the number of episodes*;
S = initial State;
while *Size(Path) < # free cells* **do**
  AddPath(S);
  Q(S,:) = Null ;
  Q(:,S) = Null ;
  S = IndexOf(min(Q(S,:))) ;
end
Result: Path

visited more than once that deteriorates the quality of the covered path.

The start position is constant (left bottom, blue 1) whereas the target position can be dynamic (red one). This means that the algorithm is able to select the last cell in order to optimize the CPP trajectory. Grey cells are known obstacles and thus cannot be visited. The optimal path occurs if all free cells are visited just once time.

Figure 2.(a) presents an environment (grid 8×8) proposed in [9] where to carry out the CPP was defined the starting and the ending point. The authors solve the CPP using a Genetic Algorithm. In this case, 71 visits were necessary to carry out the CPP, and the solution proposed in this work based on Q-learning required only 58, in other words, one visit in each free cell of the environment and does not require initial population, resulting in an easy adaptability when changing the environment.

The second case (grid 10×10) presented in Figure 2.(b) addresses the environment proposed in [4]. In order to perform the CPP, the authors used an heuristic algorithm that updates the cost of choosing the cells improving the convergence time. However, to perform the CPP of the referred environment, it is necessary to have 86 visits, while the Q-learning algorithm requires 84, one visit in each free cell of the environment.

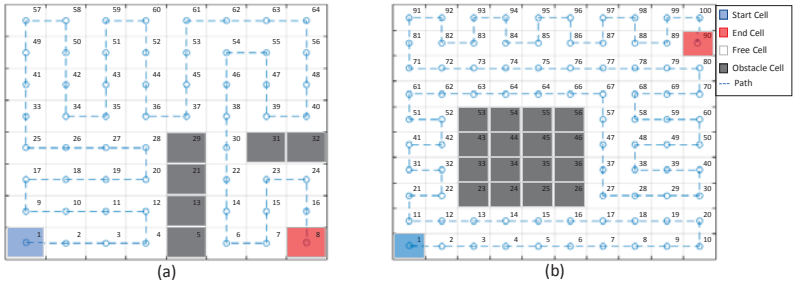

FIGURE 2. CPP based on Q-learning Algorithm: (a) Connectivity grid 8×8. (b) Connectivity grid 10×10.

Figure 3 presents the comparison for the two analyzed cases. In both situations the results of the Q-learning algorithm proposed a single visit in each free cell, optimizing the CPP for non-overlapping coverage.

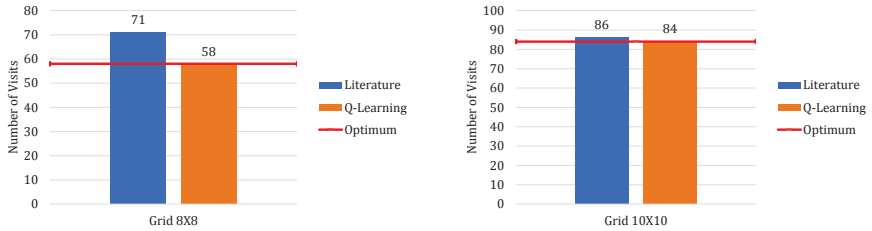

FIGURE 3. Comparison of the number of visits to perform CPP in each analyzed environment.

CONCLUSIONS AND FUTURE WORK

In this paper a Q-learning approach to solve the Coverage Path Planning (CPP) problem was presented. Unlike other applications of the Q-learning algorithm, the proposed approach tries to minimize the visited cells (overlapping) in the CPP problem. The comparison with different methodologies to solve the CPP problem validates the proposed approach. For future work the authors will applied the Q-Learning algorithm to environments with unknown obstacles.

ACKNOWLEDGMENT

This work is financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within projects SAICTPAC/0034/2015-POCI-01-0145-FEDER-016418, POCI-01-0145-FEDER-007043 and UID/CEC/00319/2013.

REFERENCES

[1] Arkin, E. M., Fekete, S. P., and Mitchell, J. S. (2000). Approximation algorithms for lawn mowing and milling. *Computational Geometry*, 17(1-2):25–50.

[2] Busoniu, L., Babuska, R., De Schutter, B., and Ernst, D. (2010). *Reinforcement learning and dynamic programming using function approximators*. CRC press.

[3] Dogru, S. and Marques, L. (2017). A*-based solution to the coverage path planning problem. In *Iberian Robotics conference*, pages 240–248. Springer.

[4] Gajjar, S., Bhadani, J., Dutta, P., and Rastogi, N. (2017). Complete coverage path planning algorithm for known 2d environment. In *Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017 2nd IEEE International Conference on*, pages 963–967. IEEE.

[5] Galceran, E. and Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276.

[6] Hameed, I. A. (2017). Coverage path planning software for autonomous robotic lawn mower using dubins' curve. In *Real-time Computing and Robotics (RCAR), 2017 IEEE International Conference on*, pages 517–522. IEEE.

[7] Hasan, K. M., Reza, K. J., et al. (2014). Path planning algorithm development for autonomous vacuum cleaner robots. In *2014 International Conference on Informatics, Electronics & Vision (ICIEV)*, pages 1–6. IEEE.

[8] Konar, A., Goswami, I., Singh, S. J., Jain, L. C., and Nagar, A. K. (2013). A deterministic improved q-learning for path planning of a mobile robot. *IEEE Trans. Systems, Man, and Cybernetics: Systems*, 43(5):1141–1153.

[9] Piardi, L., Lima, J., Pereira, A. I., and Costa, P. (2018). Path planning optimization method based on genetic algorithm for mapping toxic environment. In *International Conference on Bioinspired Methods and Their Applications*, pages 223–233. Springer.

[10] Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. PhD thesis, King's College, Cambridge.

[11] Zelinsky, A., Jarvis, R. A., Byrne, J., and Yuta, S. (1993). Planning paths of complete coverage of an unstructured environment by a mobile robot. In *Proceedings of international conference on advanced robotics*, volume 13, pages 533–538.