IMPLEMENTATION OF GRID MAPPED ROBOT PLANNING ALGORITHM IN A CONTINUOUS MAP FOR FIRE FIGHTING ROBOT

Sumarsih Condroayu Purbarani., Qurrotin A'yunina MOA., Grafika Jati., Muhammad Anwar Ma'sum,, Hanif Arif Wisesa., Wisnu Jatmiko
Faculty of Computer Science Universitas Indonesia
Kampus Baru UI Depok, Jawa Barat 16424
Indonesia

Abstract:
Fire-fighting robot is still one of the fields in robotic competitions held these days. This paper is aimed to see the implementation of the Markov Decision Planning (MDP) problem in a fire-fighting robot's navigation. The MDP algorithm evolves planning of the actions the robot should take according to the policy. This planning is mapped into a grid map. Yet in the implementation, this planning is applied in a continuous map. Using a fire-fighting robot the succession of this planning implementation is undertaken. The result shows that the implementation of grid mapped in a continuous map yields significant impacts that lead the MDP to be able to solve the limitation of wall following algorithm. This algorithm is also applied in the real autonomous mobile robot.

1. INTRODUCTION

Fire-fighting robots are still considered as one of the competed fields in robotic competitions. They are developed for finding the fires and then fighting them. Fire-fighting robot in our nowadays world is an autonomous mobile robot that should have its own control mechanism. It is including the control upon some certain constraints such as wheel slippage and rotation as we mentioned above. With precise calculation of the control mechanism robots is desired to perform any given task smoothly [1] .

In fact, the desired ideal performances are way too far. Robot often experiences several errors during their path to complete the task. Robot needs to have the required intelligence in order to complete any given tasks. Robot has to be able to know where it is or localize where it is before it does the navigation process to find the right path. It has to be able to alleviate the uncertainty that caused by the environment surrounded, avoid any collision, and also minimize the internal as well as the external errors that may occur [2] [3].

Some robots are also failed to survive from experiencing slippage while some others are failed to precisely rotate, etc. [4]. Yet, autonomous mobile robots are expected to have robust navigation system as its main prerequisite to be called "autonomous". The ability to navigate its self includes motion and path planning as well as obstacle avoidance given the built map as its environment. In addition, the inevitable uncertainty during robot's maneuver is demanded to be minimized. These uncertainties lead to the robot's localization problem which is the most common problems in robotics. In this work, we program the robot using probability approach to deal with the uncertainty explicitly.

To provide control the robot's movement is modelled by means of MDP problem. The MDP solver algorithm is then performed to help the robot find the fastest path by mapping the actions robot should take to each state. The purpose of this experiment is to implement the MDP algorithm with grid map into a continuous provided by V-Rep simulator for fire-fighting robot.

Some literatures contain various investigations about MDP. Similar to our work, the work of [4] also investigates the involvement of both discrete and continuous state variables and actions in a certain environment. The [4] experiment constructed the approximations to the optimal policy by importance sampling. Other than that, [5] purposes the qualitative and qualitatively enriched MDP algorithm. [5] reduced the problem to quadratic programming problem and then solves it using some SMT solvers. [6] also applied the similar MDP approach to motion planning for a mobile service robot. Their approach involves the method to specify tasks and synthesize cost-optimal policies using co-safe linear temporal logic.

The rest of the paper is organized as follow. In section 2, the basic of MDP is discussed briefly. It is followed by the introduction of the other implemented algorithms. In section 3 the detail of the experiment methodology is explained. In section 4 the experiment and result is elucidated and it is closured with a precise conclusion in section 5.

2. BACKGROUND

An important task for fighting robot is to find the fire and stop the spreading of fire as soon as possible. The task is modelled by MDP. The following paragraph shows the brief explanation about MDP.

MDP

MDP is commonly defined by several set of elements (S, A, T, R, γ) where S is the set of state s, A is the set of action a. As the agent performs an action, the probability of moving from state s to the new state s' can be determined

by a conditional probability function $T_{ss'}(s,a,s') = P(S_{t+1} = s'|S_t = s)$. The last two components, R, γ, denote reward function, and discount factors, respectively. The discount factor value is defined as $\gamma \in (0,1)$ to avoid the infinite value.

Table I. Value Iteration for Arbitrary Case [7]

Algorithm MDP_value_iteration():
for $i = 1$ to N do
  $\hat{V}(x_i) = r_{min}$
endfor
repeat until convergence
  for all x
    $\hat{V}(x) = \gamma \max_u [r(x,u) + \int \hat{V}(x')p(x'|u,x)dx']$
  endfor
endrepeat
return $\hat{V}$

MDP assumes that the state of the environment is fully known at all times. To find the best action (path) that leads the agent into the target, the agent computes a mapping from states to actions. Every policy for action selection, as expressed in Equation (1) is determined by the value function that will measure the cumulative discounted future payoff. The selected action is based on the optimum value function. Equation (2) shows the expected cumulative payoff.

$$\pi: x_t \rightarrow u_t \quad (1)$$

$$R_t = E\left[\sum_{\tau=1}^{T} \gamma^T r_{t+\tau}\right] \quad (2)$$

Value iteration is one of the MDPs algorithm to find the control policies. At the beginning, the value function is set to minimum value (r_{min}) and then calculated recursively for increasing horizon. Equation (3) and (4) show the value function and the policy at any point in time, respectively.

$$\hat{V}(x) \leftarrow \gamma \max_u \left[r(x,u) + \int \hat{V}(x')\, p(x'|u,x)dx'\right] \quad (3)$$

$$\pi(x) \leftarrow arg\max_u \left[r(x,u) + \int \hat{V}(x')\, p(x'|u,x)dx'\right] \quad (4)$$

Table II. Discrete value iteration for Finite Case [7]

Algorithm MDP_discrete_value_iteration():
for $i = 1$ to N do
  $\hat{V}(x_i) = r_{min}$
endfor
repeat until convergence
  for $i = 1$ to N do
    $\hat{V}(x_i) = \gamma \max_u [r(x_i,u) + \sum_{j=1}^{N} \hat{V}(x_j)p(x_j|u,x_i)]$
  endfor
endrepeat
return $\hat{V}$

The value iteration can be implemented on two different kind of states, i.e. arbitrary and finite states. For arbitrary case, the value function and the policy can be calculated using the above two equations. Meanwhile, for finite states, the mathematic notation "integral" on equation (3) and (4) is replaced by "sum" notation. Table I, II, and III depict the algorithm for arbitrary, finite states, and the policy, respectively.

Table III. Algorithm Policy [7]

Algorithm Policy($x, \hat{V}$):
return $\arg\max_u [r(x,u) + \sum_{j=1}^{N} \hat{V}(x_j)p(x_j|u,x_i)]$

Reward function

At the beginning, reward function is defined as follows:

$$R_s = \begin{cases} -0.2 & \text{If } s \text{ is not a fire source} \\ +1 & \text{If } s \text{ is a fire source} \end{cases} \quad (5)$$

The objective of this function is to distinguish the target state from another states.

3. METHODOLOGY

This experiment involves a map as seen in the Figure 1. The map is made up of four rooms which each of them has its own terminal point or checkpoint. These points are where the robot has to start observe the inside of the room whether the fire source exists or not, using the vision sensor attached to its rigid body. The robot is placed on a given initial position that is defined before the run time as well as the position of the fire source. MDP is executed before robot taking any action. Initially, each cell is set to the minimum reward value, i.e. -0.2, as shown in Figure 2.

Figure 1. Continuous map on V-Rep simulator. Green areas represent the rooms while the rest is corridor.

The output of MDP is arrows which represent each direction as actions the robot would take, as can be seen in Figure 3. The action map is drawn on a grid map on planning level. The planning is executed on the simulator map by

converting the discrete grid map into the continuous map on V-Rep simulator. The conversion is done manually by setting the range of displacement.

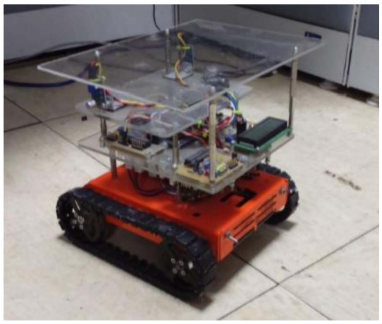


Figure 2. Al-Fath Robot

In this experiment, the positions of fire source are varied into three scenarios. In the first scenario, the fire source is placed in room 3 and the robot initial position is defined on a spot near room 3 as shown in the Figure 1. The robot has to 'build' its own action map according to the value iteration and the policy. Moreover, the robot does not observe all of the rooms at one time. Firstly, the robot is directed to its nearest room, i.e. room 3. It maps the action through MDP mechanism then executes it. Once the robot reaches the terminal point of the room, robot does actions to observe all inside the room to see if there is any fire source. There is only one fire source is placed in each scenario, thus, once the robot finds the fire source, the simulation is terminated. But if there is not any fire source, then the robot starts the MDP over again to obtain its expedition to the next rooms. This sequence continues until the fire source is found. The same process goes for the other scenarios, i.e. scenario 2 and 3 which places the fire source in room 2 and 1, respectively.

Table IV. Hardware Specification

Hardware	Number	Usage	Enabled
Atmel AT-MEGA 2560	1	Main processor	Yes
Atmel AT-MEGA 8	2	Slave processor	Yes
TGS2600	2	Odor Sensor	No
CMPS03	1	Digital Compass	No
YS1020U	1	Wireless UART	No
SRF08	2	Ultrasonic Ranger	Yes

Finally, the algorithm is also embedded to the real robot, namely Al-Fath robot [9] as can be seen in Figure 4. As shown in Table IV, Al-Fath robot uses Atmel AT-MEGA 2560 to handle most operation in the system. Furthermore, to control motor speed in left and right direction, Atmel AT-MEGA 8 is used. The robot gains information from environment using several component, for example, odor sensor, digital compass, wireless UART, and ultrasonic ranger. The brain structure of this robot is depicted in a diagram in Figure 3.

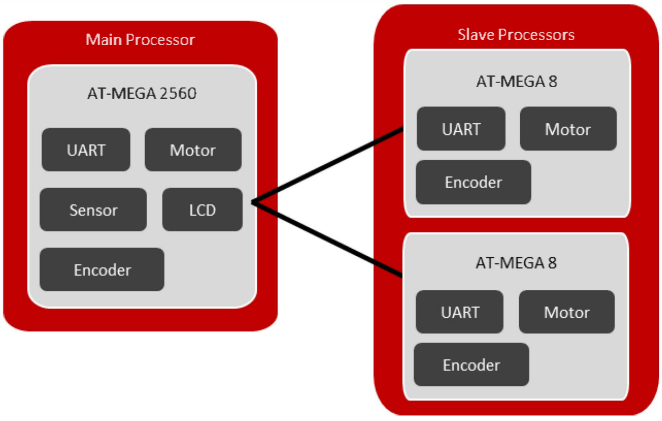


Figure 3. Al-Fath Brain Structure

To see how good this planning algorithm performs, the robot is also tested using wall-following algorithm.

4. Result and Discussion

The output of MDP algorithm is the value function which defines the rewards value. Based on this value, the direction to the goal can be generated. Figure 4 (a) presents the reward value for each states. Meanwhile, Figure (b) shows the directions. There are some symbols used to define the motion in this experiment. The detail of the symbol can be explained in Table V. For scenario 2 as depicted in Figure 4, the best actions consists of "^, a, >, >, >". As previous mentioned, the best action is selected based on the maximum cumulative reward value. The total reward of this motion is about 1.370.

Table V. Symbol definition

Symbol	Action
^	Move to north direction
>	Move to east direction
<	Move to west direction
v	Move to south direction
a	Move to northeast direction
b	Move to northwest direction
c	Move to southwest direction
d	Move to southeast direction

According to Figure 5, the experiments that used Wall following algorithm consume less time than the MDP one for the scenario 1 and 2. This is due to the conversion of the grid map to the continuous map took place in the planning approach. The conversion causes every action in the continuous map restricted by some certain range. Therefore, the robot could not move freely as it is when using a non-planned map, in this case it is wall-following algorithm. Yet, in scenario 3, the wall-following robot failed to reach the fire

source. This is due to the plain wall-following mechanism which only evolves the right-wall-following and left-wall-following. Another words, this approach could only find the targets on the path near walls.

-1.469	-1.302	-1.203	-1.105	-1.108	nan	-0.787	-0.754	-0.744	-0.764
-1.385	-1.238	-1.050	-0.966	-0.855	nan	-0.587	-0.543	-0.520	-0.549
-1.344	-1.149	-1.002	-0.784	-0.720	-0.604	-0.392	-0.327	-0.293	-0.315
-1.317	-1.114	-0.899	-0.768	-0.505	-0.485	-0.379	-0.103	-0.063	-0.068
-1.306	-1.089	-0.871	-0.640	nan	-0.210	nan	nan	0.193	0.182
-1.317	-1.115	-0.876	-0.635	-0.355	-0.092	0.081	0.365	0.425	0.438
-1.354	-1.154	nan	nan	nan	nan	0.171	nan	0.676	0.699
-1.402	-1.413	nan	-0.589	-0.309	-0.056	0.172	0.464	0.698	1.000
-1.409	-1.185	nan	nan	nan	-0.092	0.159	nan	0.676	0.699
-1.408	-1.172	-0.899	-0.624	-0.344	-0.123	-0.123	nan	0.438	0.438

(a)

d	d	d	d	v	None	d	d	v	c
d	d	d	d	d	None	d	d	v	c
d	d	d	d	d	d	d	d	v	v
d	d	d	>	d	v	>	d	v	v
>	>	>	d	None	d	None	None	v	v
a	>	>	>	>	d	>	d	d	v
a	a	None	None	None	None	d	None	d	v
a	^	None	>	>	>	>	>	>	.
d	d	None	None	None	a	a	None	a	^
>	>	>	>	a	a	^	None	a	^

(b)

Figure 4. The reward value (a) and the direction (b) of motion in Scenario 2; blue and red point represent the starting and the final point, respectively

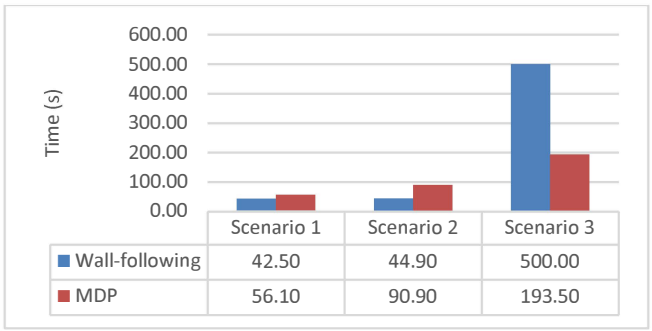


Figure 5. Total Elapsed Time to Reach Each Fire Source

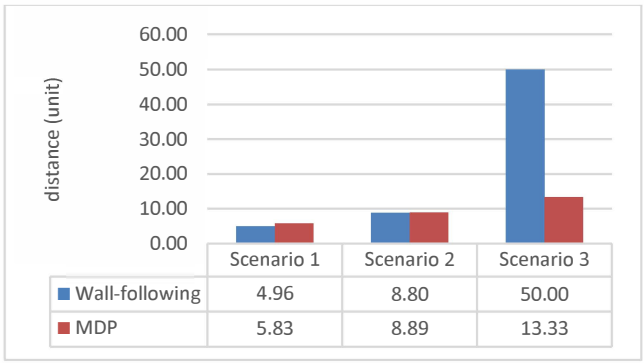
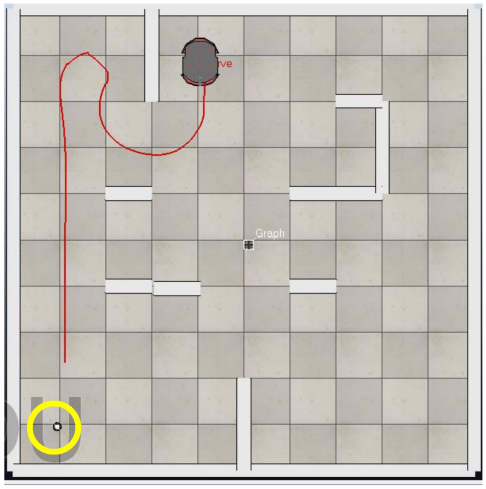
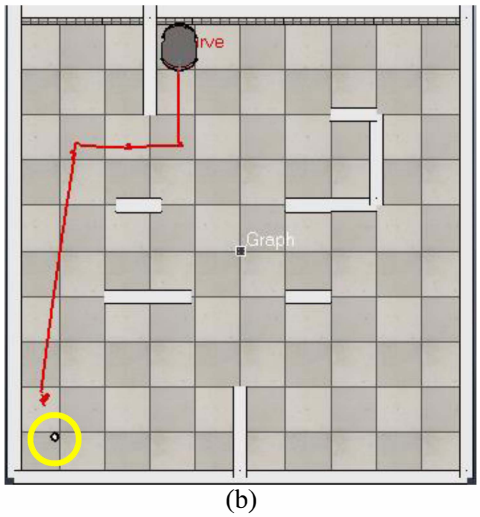


Figure 6. Total Path Length Taken to Reach Each Fire Source

Furthermore, Figure 6 shows the trajectory distance experienced by the robot. In the scenario 1, the wall-following robot has shorter distance than the planning robot has. Moreover, in the scenario 2, the planning robot also experienced much longer path than the other robot. This phenomenon can be explained by the fact that the fire source was placed near the wall which can be easily detected by the wall following robot as it passed the trajectory. It can be clearly seen in Figure 7 that shows how the implementation of MDP involved a shorter trajectory compared with the one that wall-following algorithm took. The red line depicted in this figure represents the taken path. Surprisingly, in scenario 3, the wall following could not reach the fire source since it is way far from the wall that on its following path. Thus, it generated a longer path than the MDP robot. The path is depicted in Figure 8.

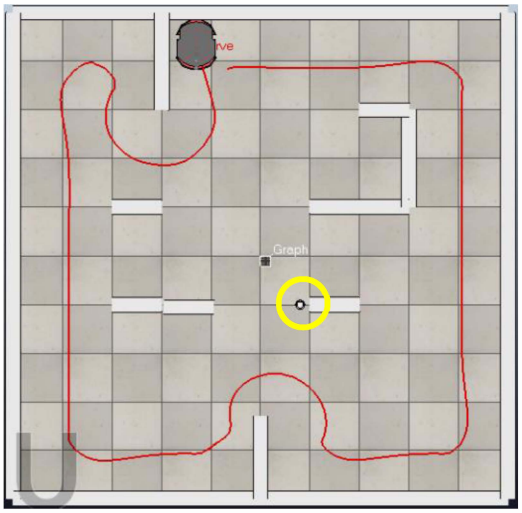


(a)

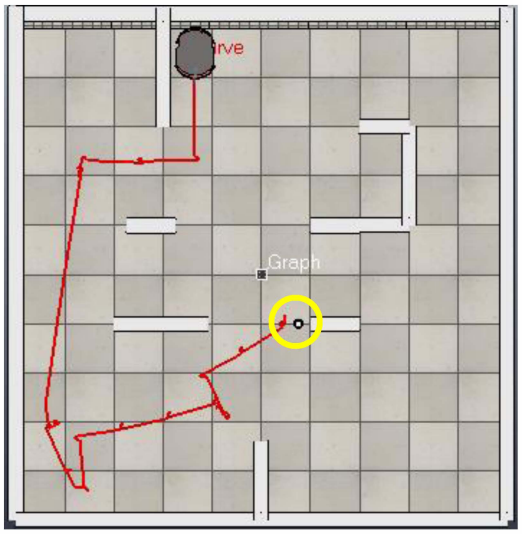


(b)

Figure 7. Fire position placed in room 2 (scenario 2). Yellow circled object on the map represents the fire source. (a) Wall Following Case; (b) MDP Case



(a)



(b)

Figure 8. Fire position placed in room 1 (scenario 3). Yellow circled object on the map represents the fire source. (a) Wall-Following Case; (b) MDP Case

5. Conclusion

This paper described the implementation of Markov Decision Process (MDP) for solving a fire fighting robot's navigation and also demonstrated how to apply it to planning in a real robot, namely Al-Fath. To test the robustness of the algorithm, the robot was tested using three scenarios. The performance was compared with the wall following algorithm mechanism. The result shows that the planning robot has a better performance than the non-planning robot only when the fire source placed far from the wall on the wall following robots trajectory. Nonetheless, for the distance parameter, the planning robot has longer distance path since the robot cannot move freely due to the range on each step yielded by the grid action planning map.

6. Acknowledgement

This research work was supported by national research funding in 2015 "Penelitian Unggulan Perguruan Tinggi Negeri (PUPTN)". The title of the research is intelligent traffic system for sustainable environment. This grand number is 0476/UN2.R12/HKP.05.00/2015.

References

[1] Ocaña, M., Llamazares, Á., Molinos, E., Hernández, N., Herranz, F., Revenga, P., & López, E., "Development of a Navigation System for a Robotic Shop Guide," *sensors,* vol. 11, 2014.

[2] Kummerle, Rainer and Ruhnke, Michael and Steder, Bastian and Stachniss, Cyrill and Burgard, Wolfram, "A navigation system for robots operating in crowded urban environments," in *2013 IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[3] R. Kümmerle, "State Estimation and Optimization for Mobile Robot Navigation," Universität Freiburg , 2013.

[4] Siegwart, Roland and Nourbakhsh, Illah Reza and Scaramuzza, Davide, Introduction to autonomous mobile robots}, MIT Press, 2011.

[5] Nitti, Davide and Belle, Vaishak and De Raedt, Luc, "Planning in discrete and continuous Markov decision processes by probabilistic programming," in *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2015.

[6] Li, Meilun and She, Zhikun and Turrini, Andrea and Zhang, Lijun, "Preference Planning for Markov Decision Processes," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* , 2015.

[7] Lacerda, Bruno and Parker, Dennis and Hawes, Nick, "Optimal and dynamic planning for Markov decision processes with co-safe LTL specifications," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, 2014.

[8] Sebastian Thrun, Dieter Fox, Wolfram Burgard, Probabilistic Robotics, 1999.

[9] D. Fox, "KLD-sampling: Adaptive particle filters," in *Advances in neural information processing systems*, 2001.

[10] Röwekämper, J., Sprunk, C., Tipaldi, G. D., Stachniss, C., Pfaff, P., & Burgard, W., "On the Position Accuracy of Mobile Robot Localization based on

Particle Filters Combined with Scan Matching," in *Intelligent Robots and Systems (IROS)*, 2012.

[11] H. Kurniawati and V. Yadav, "An Online POMDP Solver for Uncertainty Planning in Dynamic Environment," in *ISSR*, 2013.

[12] J. Hoey, A. Von Bertoldi, P. Poupart and A. Mihailidis, "Assisting persons with dementia during handwashing using a partially observable Markov decision process," in *International Conference on Vision Systems*, 2007.

[13] S. Thrun, "Monte Carlo POMDPs," in *NIPS*, 1999.

[14] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," in *Advances in neural information processing systems*, 2010.

IMPLEMENTATION OF GRID MAPPED ROBOT PLANNING ALGORITHM IN A CONTINUOUS MAP FOR FIRE FIGHTING ROBOT

Sumarsih Condroayu Purbarani., Qurrotin A'yunina MOA., Grafika Jati., Muhammad Anwar Ma'sum,, Hanif Arif Wisesa., Wisnu Jatmiko
Faculty of Computer Science Universitas Indonesia
Kampus Baru UI Depok, Jawa Barat 16424
Indonesia

Fire-fighting robot is still one of the fields in robotic competitions held these days. This paper is aimed to see the implementation of the Markov Decision Planning (MDP) problem in a fire-fighting robot's navigation. The MDP algorithm evolves planning of the actions the robot should take according to the policy. This planning is mapped into a grid map. Yet in the implementation, this planning is applied in a continuous map. Using a fire-fighting robot the succession of this planning implementation is undertaken. The result shows that the implementation of grid mapped in a continuous map yields significant impacts that lead the MDP to be able to solve the limitation of wall following algorithm. This algorithm is also applied in the real autonomous mobile robot.