Multi-Agent Path Planning With Dynamic Obstacles in a Partially Known Environment

Rishab Shah
Boston University
8 St. Mary's Street Boston MA 02215
rishah@bu.edu

Abstract

We propose an efficient variant of the D lite algorithm for planning in partially known maps and environments with dynamic obstacles and extend it to a multi-robot system. We make this into a motion planning problem rather than a coverage problem which is common with multi-agent systems. We find ourselves at a crossroad in selecting the mode of communication between the robots with a centralized or a decentralized, distributed approach. A distributed approach is used for communication between the agents as it allows for lower complexity and better planning.*
It is essential for robust planning to take into account the inherent uncertainty with the obstacles in the problem, which arises due to uncertain localization, modeling errors, and disturbances. Prior work has handled the case of deterministically bounded uncertainty. We propose an alternative approach that uses a probabilistic representation of uncertainty for modelling obstacles with additive Gaussian noise that has known statistics.
A simulation scenario will reflect the experimentation for this problem. The results would be evaluated based on optimality, which means shortest path with no collisions.

1. Introduction

Robots operating in an unknown, partially known and/or dynamic environment that are required to perform a given task are faced with the problem of deciding where to go to get the most relevant information for their current task. We are particularly interested in the simplest task in a path planning problem, i.e., finding an optimal path for all the robots between two locations.

When the environment is known, the path planning problem consists of computing paths that allow the robots to sense the regions of interest in an optimal way according to some metric. However, when the environment is unknown or partially known, the robots must also learn the structure of the environment by identifying which regions are interesting and which are not. Hence, these paths are where the most sensory information can be obtained by the robots.

Given a directed or undirected graph and a set of agents with unique start and goal vertices, the Multi-Agent Path Finding (MAPF) problem is to find collision-free paths for all agents from their respective start vertices to their respective goal vertices [3]. Minimizing the solution cost given by the sum of the travel times of the agents along their paths is NP-hard [22].

Algorithms that are commonly used in partially known environments with dynamic obstacles are D* Lite and variants like Focused D*, Moving Target D* Lite (which is primarily used for tasks with moving targets). The actual performance of D* Lite is dependent upon the size of the grid, number of node expansions and heap sorting. D* Lite is free from the drawbacks like converging to local minimums as is the case with Potential-field algorithms.

In this paper, we present a novel implementation of the D* Lite algorithm applied to a multi-robot system where the obstacles are modeled with uncertainty as a Gaussian distribution as the robots traverse the environment towards their goal. To consider all possibilities, we look at the case when the robots have different start and end locations. We exploit the fact that the robots have a limited field of view around them and build a probability distribution of the obstacles that every other robot updates their map with as they move toward the goal.

2. Background and Related work

2.1. Different applications

Multi-agent systems are often adopted to accomplish tasks where cooperation between different robots is preferred, like situations that are hazardous or time and power consuming for a single agent. Extensive work has already been done [4] [20] [24] [18] in the application areas such as , factory floor robots, area reconnaissance, task reassignment in multi-robot teams, cleaning robots etc. Traditional

path planning methods such as Visibility graph, Free space method, Grid method, Topological method and Potential-field method offer great success in multi-robot path planning.

Path finding is also a component of many video games. For example, agents in turn-based or real-time strategy games need to plan collision-free paths from their current locations to their goal locations, often in dynamic and congested environments. It is easier for players to control hundreds of agents by moving the agents in a team rather than individually. MAPF can be solved via reductions to other well-studied problems [17] [21] [6] or by optimal or suboptimal MAPF algorithms [15] [19]. Many MAPF algorithms have been used on maps from video games [13].

Multi-agent systems are also employed in both the military and the civilian domain for surveillance missions with a system of multiple Unmanned Aircraft by means of a Kalman Filter technique [8]. They are slightly different from the strategy we plan to use as they are based on the development of a target tracking algorithm to provide information on both target and drones motion on the surveillance area by means of a Kalman Filter and Bayesian network.

Search and Rescue operations are also performed by adopting the multi-agent systems and they may be generally characterized through multiple dimensions and attributes including: stationary vs. moving target search, discrete vs. continuous time and space search, static/dynamic as well as open and closed-loop decision models [2].

2.2. Path planning algorithms

Focused D* [16], and D*Lite [9] are popular replanning algorithms amenable for path-planning in partially known terrain. D*Lite extensions are used in the navigation algorithms for a Mars Rover [7], and in the DARPA Urban challenge competition [11]. [Making A* Run Faster than D*-Lite for Path-Planning in Partially Known Terrain]

When the agent has partial knowledge of the terrain we make a free space assumption [23] which is an optimistic assumption about the grid in which unknown cells are considered obstacle-free. Adaptive A* (AA*) [10] is a replanning algorithm that uses A* to compute a complete solution each time one is needed.

In applications where robots have to continually monitor or sense regions of interest for a long period of time, the robots generate closed paths that allow them to sense regions of interest. These paths are referred as Interesting Closed Paths (IC paths). The robots could then travel these closed paths for long periods of time until the monitoring task is done. A wide range of applications require robots to move along IC paths in unknown environments in order to concentrate the robots resources in the regions of interest and avoid wasting them in non-interesting regions. The algorithm presented in [14] allows the user to have liberty of controlling the robots speed along the IC paths.

2.3. Planning under uncertainty

Another paper [12] uses the uncertainty of the mapped obstacles in the planning stage to minimize the chance of collision when navigating. This method fundamentally bases on Rapidly-Exploring Random Trees (RRTs) and inherits the sampling-based planners sub-optimality and probabilistic completeness.

Newer approaches like [5] [1] involve stochastic robot observations and dynamics: after applying a motion command, the actual robot position differs from the predicted one, due to actuation noise. Even more important, the robot does not know, a priori, which measurements will be acquired and what the (future) sensor readings will be.

3. Problem Definition

3.1. Description of the problem

We assume that a single obstacle map is available for all the agents involved in performing the given task. The map is available in shape of a graph $G = (V, O, c)$, where V is the set of vertices, O is the set of arcs and c is a cost function $c : O \rightarrow \mathbb{R}$. This graph represents the connectivity of both free and occupied space around multiple robots involved.

We define a vertex within graph G as $V_{x,y} = (r_{id}, t)$. Here x and y are the index of the vertex on grid-like eight connected graph and R_{id} is the robot id. Variable is set to null (ϕ) for some vertex that is not part of a planned path for any of the robots. We assume that a set consists of robot identities and is defined by $R = \{r_1, r_2, \ldots r_k\}$ represents k robots (agents) which share the same obstacle map. We also assume that collisions occur when a robot r_i tries to enter a vertex which has the value of its $t \neq 0$. Each robot should ideally follow a collision-free path in order to reach its predefined goal.

3.2. Assumptions

The robots from set R have a predefined set of goals $E = \{e_1, e_2, \ldots e_k\}$ with each of their start positions also defined as $S = \{s_1, s_2, \ldots s_k\}$. A limitation of the D* Lite algorithm is that the positions of the goal cannot be changed until all robots have reached their destinations. In this paper, we assume that each robot perceives any other robot as a dynamic obstacle and not to take advantage of the spatiotemporal nature of obstacles as all robots share the memory and know the path of the other robots.

Now the algorithm is expected to compute feasible collision-free paths for all the robots defined by set R. All robot paths must start at start positions defined by set S and end at goal positions defined by set E.

A heuristic function h for a path-planning problem is a non-negative function such that *h(s)* estimates the cost of a

path from s_i to e_i . Function h is admissible if for every state s_i, $h(s_i)$ does not overestimate the cost of any path from s_i to e_i . Furthermore, we say h is consistent if for every *(s, t)* $\in O$ it holds that $h(s_i)c(s_i,t)+h(t)$, and $h(e_i)=0$.

4. Experiments and future

For a multi-agent system, each robot needs to communicate to each other their start positions and the information in their field of view. We assume a partially known map, and hence the robots are aware of the goal location. We use a shared memory that acts as a central server for sharing the information about the map and updating the path. We assume all the robots move at the same, constant speed initially and plan to extend this to different constant speeds. The obstacles are initially modeled to be deterministic to reduce the complexity. As for the uncertainty, each robot later models the environment with a given amount of probabilities of obstacles provided beforehand. Then, as they begin to traverse the path, we update the uncertainties based on what the robot senses (here, artificially) in its view.

We use Mobile Robot Programming Toolkit (MRPT) to perform the simulation. OpenGL packages will employed to obtain a visual representation of the problem. Currently, we have implemented the D* Lite algorithm with dynamic obstacles for a single robot where it has a limited filed of view and the obstacles are deterministically modeled. There is no visual representation currently available. The two dimensional world in question is of polygonal nature and the robot itself occupies a certain area. Planning is more complicated by the fact that the robot also occupies a non-trivial area as it has to maneuver its body around obstacles. In our implementation, we use a square shape for the robot of size = 4*unit size. The units of occupancy grid are mapped to nodes in the eight-connected obstacle-map graph G, thus converting physical adjacency relationships to graph connectivity. We use the minimum Euclidean distance to be the priority heuristic. The obstacles are also assumed to move at a constant speed randomly and can also appear without existing before. We limit the latter feature so that the new obstacles appear at most 4 times of maximum size 6*unit size anywhere in the map to avoid congesting the map too much.

4.1. Challenges

We are working on extending the current implementation to a multi-agent system. Some of the challenges faced are: the shared map does not update with all agents as we need to update the map after each step of each of the robots, modelling the obstacles into a Gaussian probability distribution does not update when the obstacle is in the field of view of any robot, placing the obstacles that are dynamic is a problem as cases when a dynamic obstacle tries to occupy a cell that is already currently occupied, crashes the program, this can be solved by defining rules for the obstacles to follow and the speed of running the program is slowing down considerably for the multi-agent case. Since the execution time of D* Lite is dependent upon the frequency of change in obstacles, multi-robot D* Lite adds to the time complexity of D* Lite by a polynomial factor. This factor is directly proportional to the number of robots (k) involved in multi-robot path planning. The factor is also dependent upon the probability of path cross over probability of obstacle change.

References

[1] T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.

[2] J. Berger and N. Lo. An innovative multi-agent search-and-rescue path planning approach. *Computers & Operations Research*, 53:24–31, 2015.

[3] L. Cohen, T. Uras, T. K. S. Kumar, H. Xu, N. Ayanian, and S. Koenig. Improved solvers for bounded-suboptimal multi-agent path finding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 3067–3074. AAAI Press, 2016.

[4] R. N. De Carvalho, H. Vidal, P. Vieira, and M. Ribeiro. Complete coverage path planning and guidance for cleaning robots. In *Industrial Electronics, 1997. ISIE'97., Proceedings of the IEEE International Symposium on*, volume 2, pages 677–682. IEEE, 1997.

[5] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[6] E. Erdem, D. G. Kisa, U. Öztok, and P. Schüller. A general formal framework for pathfinding problems with multiple agents. In *AAAI*, 2013.

[7] D. Ferguson and A. Stentz. Field d*: An interpolation-based path planner and replanner. *Robotics research*, pages 239–253, 2007.

[8] D. Gentilini, N. Farina, E. Franco, A. E. Tirri, D. Accardo, R. S. L. Moriello, and L. Angrisani. Multi agent path planning strategies based on kalman filter for surveillance missions. In *Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI), 2016 IEEE 2nd International Forum on*, pages 1–6. IEEE, 2016.

[9] S. Koenig and M. Likhachev. Fast replanning for navigation in unknown terrain. *IEEE Transactions on Robotics*, 21(3):354–363, 2005.

[10] S. Koenig and M. Likhachev. A new principle for incremental heuristic search: Theoretical results. In *ICAPS*, pages 402–405, 2006.

[11] M. Likhachev, D. I. Ferguson, G. J. Gordon, A. Stentz, and S. Thrun. Anytime dynamic a*: An anytime, replanning algorithm. In *ICAPS*, pages 262–271, 2005.

[12] K. Ok, S. Ansari, B. Gallagher, W. Sica, F. Dellaert, and M. Stilman. Path planning with uncertainty: Voronoi uncertainty fields. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4596–4601. IEEE, 2013.

[13] D. Silver. Cooperative pathfinding. *AIIDE*, 1:117–122, 2005.

[14] D. E. Soltero, M. Schwager, and D. Rus. Decentralized path planning for coverage tasks using gradient descent adaptive control. *The International Journal of Robotics Research*, 33(3):401–425, 2014.

[15] T. Standley and R. Korf. Complete algorithms for cooperative pathfinding problems. In *IJCAI*, pages 668–673, 2011.

[16] A. Stentz et al. The focussed dˆ* algorithm for real-time replanning. In *IJCAI*, volume 95, pages 1652–1659, 1995.

[17] P. Surynek. Reduced time-expansion graphs and goal decomposition for solving cooperative path finding sub-optimally. In *IJCAI*, pages 1916–1922, 2015.

[18] M. Takahashi, T. Suzuki, H. Shitamoto, T. Moriguchi, and K. Yoshida. Developing a mobile robot for transport applications in the hospital domain. *Robotics and Autonomous Systems*, 58(7):889–899, 2010.

[19] K.-H. C. Wang and A. Botea. Mapp: a scalable multi-agent path planning algorithm with tractability and completeness guarantees. *Journal of Artificial Intelligence Research*, 42:55–90, 2011.

[20] E. K. Xidias, A. C. Nearchou, and N. A. Aspragathos. Vehicle scheduling in 2d shop floor environments. *Industrial Robot: An International Journal*, 36(2):176–183, 2009.

[21] J. Yu and S. M. LaValle. Planning optimal paths for multiple robots on graphs. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3612–3617. IEEE, 2013.

[22] J. Yu and S. M. LaValle. Structure and intractability of optimal multi-robot path planning on graphs. In *AAAI*, 2013.

[23] A. Zelinsky. A mobile robot exploration algorithm. *IEEE Transactions on Robotics and Automation*, 8(6):707–717, 1992.

[24] R. Zlot and A. Stentz. Market-based multirobot coordination using task abstraction. In *Field and Service Robotics*, pages 167–177. Springer, 2003.