Towards a Robot Path Planner for Dangerzones and Desirezones

Z. M. Ma[1], Froduald Kabanza[1,*], Khaled Belghith[2], and Roger Nkambou[2]
[1]Département d'informatique, Université de Sherbrooke
Sherbrooke, Québec J1K 2R1, Canada
zongmin.ma@usherbrooke.ca, kabanza@usherbrooke.ca
[2]Département d'informatique, Université du Québec A Montréal
Montréal, Québec, H3C 3P8, Canada
belghith.khaled@courrier.uqam.ca, nkambou.roger@uqam.ca

ABSTRACT
This paper discusses the problem of planning a robot path in a workplace that contains a set of *obstacles*, a set of *dangerzones*, and a set of *desirezones*. The path cannot collide with the obstacles, should avoid the dangerzones, and should go through the desirezones. In order to plan such a path, we introduce the notion of *fuzzy probabilistic roadmap* (*fuzzy PRM*) as an extension of the traditional PRM. Each edge in a fuzzy PRM is assigned a possibility degree to represent the possibility that it is feasible. We also introduce an algorithm for computing a fuzzy PRM that uses lazy collision checking; that is, the collision of edges in the fuzzy roadmap with obstacles is delayed until a candidate path has been computed.

KEY WORDS
Robot path planning, dangerzones, desirezones.

1. Introduction

Path planning has been extensively studied in robotics over the past two decades and is increasingly becoming important in areas such as computer animation, computer aided design, and medical robotics equipments (e.g., for surgery). In its traditional form, the path planning problem is to plan the path for a moving body (typically a robot) from a given start configuration to a given goal configuration in a workspace containing a set of obstacles. The basic constraint on solution paths is to avoid collision with obstacles, which we call hereby a *hard constraint*. There exist numerous strategies for path planning under this constraint [1, 2].

In many complex applications, however, in addition to obstacles that must be avoided, we may have areas that must be avoided as much as possible. That is, a path going through these areas is not highly desirable, but would be acceptable if no better path exists or can be computed efficiently. The danger concept is relevant for example in military applications and path planning techniques to deal with it have been proposed, including [3, 4]. Dually, it may be desirable for a path to stay close to certain areas as much as possible. We call such constraints on desirable zones and dangerzones (or undesirable ones) *soft constraints* on the robot path. Compared with the hard constraints, soft constraints are not mandatory; they just provide criteria for rating solutions paths which must avoid obstacles, that is, the more a path avoid dangerzones and goes through desirezones, the better it is. This means soft constraints may be broken, for example by trading the path quality against the cost of computing it.

In this paper, we discuss one idea for extending a traditional PRM planning approach to deal with a set of *dangerzones*, and a set of *desirezones*. More specifically, we propose a new path planner that builds fuzzy roadmaps, by extending existing techniques for delay collision checks, single query, bi-direction, and adaptive sampling [5]. Our algorithm is still under implementation, so the discussion in this paper is on the algorithm description and the ideas behind it, reserving simulations and benchmarks to a follow up paper.

The remainder of this paper is organized as follows. In Section 2, we introduce the concepts of *desirezones* and *dangerzones* as well as the notion of fuzzy PRM. Section 3 explains one approach for extending an existing PRM planning method to deal with fuzzy PRM. A brief overview of related work is given in Section 4. Section 5 concludes this paper and discusses our future work.

2. Path Planning

2.1 Probabilistic Roadmap (PRM) Planner

Path planning approaches can globally be subdivided into three classes: cell-decomposition, roadmap, and potential field [2]. Among roadmap approaches, the probabilistic roadmap planner (PRM) is a relatively new technique builds a roadmap by randomly sampling the robot configuration space (C-space), rather than using deterministic search techniques.

* Corresponding author.

In its original formulation [6], a PRM planner builds a roadmap of nodes in C-space, by picking nodes randomly and checking that they do not collide with obstacles, using a local planner to check that edges between two adjacent nodes is also collision-free. Each time a local planner succeeds, the corresponding edge (local paths) is inserted into the roadmap. Building the roadmap is called the *learning phase*. Upon the built roadmap, individual planning queries are solved by adding the start and goal configuration as nodes in the roadmap and then using graph search to find a path connecting these nodes. Using the roadmap to look for the final solution path is called the *query phase*. It is not difficult to find out that the traditional PRM mainly follows the followings stages: constructing a collision-checked roadmap (excluding start and goal configurations) and searching the roadmap for a shortest path (no collision check needed). Having the pre-computed roadmap without the start and goal configurations, the traditional PRM is very suitable for multiple queries. For each query, the given start and goal configurations are connected to the roadmap and then a shortest path is searched for.

It has later been shown that PRM planners spend most of their time performing collision checks for edges between nodes [5], while many of them end up being ruled out from the final path. This is one argument against pre-computing a complete roadmap. Another argument is when we have dynamically changing configuration spaces (this is the case, for example, when the robot often changes tools, grasps objects, or new obstacles enter the workspace). In such case, each query requires a different roadmap. These two arguments prompted the some authors to introduce new PRM methods that delay collision tests between nodes until they are absolutely needed [5, 7].

These lazy PRM approaches still follow the same basic PRM approach, that is: constructing a collision-checked roadmap and searching the roadmap for a shortest path. The two only difference are that (a) this time the initial and the goal configuration are in the PRM and (b) no local planner is this time involved until a path from the initial configuration to the goal configuration has been computed; such a path is a sequence of nodes in the PRM, such that each node has been checked to be in the free C-space, but the connection between two consecutives has not yet been checked for the absence of collision. Checking paths for collision is done when a shortest path is found, hence justifying the name "lazy PRM". It is clear that the lazy PRM is particularly suited for single queries.

2.2 Fuzzy PRM for Dangerzones and Desirezones

In the above setting, the only requirement for a path is to avoid obstacles. Our approach allows the option to add constraints of *staying away from some obstacles and/or some dangerzones as much as possible* and *being close to some desirezones as much as possible*. More specifically, a moving robot is valid if the robot does not collide with any obstacles and if the path satisfies some soft constraints on the possible motions for the robot as follows:

(1) Staying away from the obstacles;
(2) Avoiding the dangerzones as much as possible;
(3) Going through the desirezones as much as possible;

Note that condition (2) can be formulated to convey the constraint of staying away from dangerzones since the notion of going through is defined in our approach with a degree of possibility. Similarly, condition (3) can be formulated in such a way to constrain the path to stay close to desirezones as much as possible.

In a traditional PRM, an edge between two nodes represents that fact that a local planner can connect the two nodes by feasible path. In other words, we could be said that this edge, called *local path*, is feasible with possibility degree 1.0. The absence of an edge means that the connection either fails or is never tried. That is, the absent edge is feasible with possibility degree 0.0.

In our fuzzy PRM, edges are feasible with possibilities between 0 and 1, the edges with possibility 0 (in collision) being a particular case at the lower extreme grading of desirabilities and the edge with possibility 1 (completely in free C-Space, away from dangerzones and fully in most desirable zones) being on the other extreme. Based on fuzzy set and possibility distribution theories [8, 9], each edge of the roadmap is associated with a possibility degree, called the *edge possibility*, and such a roadmap is named *fuzzy roadmap*.

The possibility degree associated with an edge in the fuzzy PRM is an estimate of the chance that the edge is actually feasible, i.e., satisfaction of the constraints. In the flexible path planning, the workplace contains a set of *obstacles*, a set of *dangerzones*, and a set of *desirezones*. The nodes of edges can be identified to overlap with obstacles, danger zones, common zones, or desire zones. Basically an edge possibility is mainly determined by where its two nodes are located. How locations of two nodes influence the possibility of the edge connecting the two nodes will be illustrated in Section 3.2.

Using the fuzzy PRM, we can find the most possible path from start configuration to goal configuration through the fuzzy PRM. Then the actual verification of the path is handled.

3. Description of Fuzzy PRM Planner

Our fuzzy PRM planner is given two parameters, which are the maximum number of nodes that it is allowed to generate and a distance threshold for determining if two

configurations are considered "close" to one another. In what follows, these two parameters are denoted by *max* and ρ, respectively.

3.1 Overall Algorithm

Following the steps in [5] to search for a path from the start configuration to the goal configuration, our fuzzy PRM planning algorithm consists in building two trees rooted at the start and goal configurations, respectively. As soon as the two trees intersect, a feasible path can be extracted. Then the path is checked for collision. Here the fuzzy PRM planner does not immediately test connection between nodes for collision when the trees are constructed. Only when a sequence of nodes joining the start and goal configurations is found, the connections between nodes along this path are tested.

Algorithm FUZZ_PLANNER (q_i ,q_g, `obstacles`, `robot`)
Step 1: Install start and goal configurations q_i and q_g as the roots of T_i and T_g, respectively
Step 2: Repeat *max* times
  Step 2.1: EXPAND-TREE
  Step 2.2: $\tau \leftarrow$ CONNECT-TREE
  Step 2.3: if $\tau \neq$ nil then $\tau \leftarrow$ TEST-TREE
  Step 2.4: if $\tau \neq$ nil then return τ
Step 3: Return *failure*

The input consists of an initial configuration, q_i, a goal configuration q_g, a list of obstacles, and the robot kinematics. The planner builds two trees of nodes, T_i and T_g, respectively rooted at the start and goal configurations q_i and q_g. Initially, T_i contains just one node, q_i, whereas T_g contains just q_g. At step 2.1, EXPAND-TREE iteratively expands both trees by randomly selecting one of the trees, then randomly choosing a current node to expand in the selected tree, then a randomly generating successor node that is in the free C-space and in the close neighbourhood of the current node (i.e., within a distance less than a given ρ) and creating a segment between the two; this process goes on until both trees have nodes that are close so they can be connected to join them (Step 2.2). Once both trees are connected, this means that we have a candidate path, but the segments joining its nodes have not yet been tested for the absence of collision. Step 2.3 makes the required test; if successful the candidate path is returned as a solution; otherwise, Step 2 is resumed up to a fixed number of iterations or until finding a solution. The planner returns *failure* if it has not found a solution path after *max* iterations. At this moment, either no collision-free path exists between q_i and q_g, or the planner actually failed to find one.

The above is an abstract description of the original PRM planner in [5]. Our fuzzy PRM planner has the abstract description, the fundamental differences being in the details of Step 2.1 (EXPAND-TREE) and Step 2.2 (CONNECT-TREE) and the characterization of the returned solution path. In our case, based on their overlap with dangerzone or desirezones, configuration nodes have degrees of possibilities; these degrees are taken into account when choosing a node to expand in Step 2.1 and when connecting the two trees in Step 2.2. At the end, we also use the possibility measure to return a path associated with its degree of possibility.

3.2 Tree Expansion

When expanding a tree T (either T_i or T_g), the first question to consider is:

which node q should be picked from T for expansion.

The second question is:

which node q_{new} should be picked from the free C-space, not already in T, so as to add it to T by creating a segment between q and q_{new}?

Our fuzzy PRM planning approach is to *pick a node q from T with highest possibility distribution Π (q)*. Here the possibility distribution $\Pi(q)$ of m is defined as the product of the possibility of the edges e_1, e_2, …, e_n along the path from the root of T to node q.

In the original PRM planning approach, q_{new} has to be picked from in the C-space in the neighbourhood of q. In addition, our approach requires that the edge connecting q and q_{new} should have a high edge possibility. Note that the collision test of the segment from m to q is not done at this point; this is done later only for segments that are found to belong to candidate paths. Now let us focus on how to compute the possibility of an edge.

Given a segment connecting two nodes, we have the following cases:

Case 0: one node overlaps with an obstacle or the segment is on collision;
Case 1: both nodes overlap with the dangerzones;
Case 2: one node overlaps with a danger zone and another with a common zone;
Case3: one node overlaps with the dangerzones and another with a desire zone;
Case 4: both nodes overlap with the common zones;
Case 5: one node overlaps with a common zone and another with a desire zone;

Then we have seven kinds of edge listed above. It is clear that they have different edge status: the possibilities from *Case 0* to case 6 are increased successively. The possibility in case 0 is the least and the possibility in *Case 6* is the greatest. We can use values in [0, 1] to these edge possibility. For example, we may use 0.0, 0.2, 0.3, 0.4, 0.6, 0.8, and 1.0 to represent the possibilities from *Case 0* to *Case 6*, respectively.

Algorithm EXPAND_TREE
Step 1: Pick T to be either T_i, or T_g, each with probability 1/2
Step 2: Repeat until a new node q has been generated
  Step 2.1: Pick a node q from T with highest possibility distribution $\Pi(q)$
  Step 2.2: For i = 1, 2, …until a new node q has been generated
    Step 2.2.1: Pick a configuration q_{new} uniformly at random at distance less than ρ from m
    Step 2.2.2: If q_{new} is collision-free and the edge connecting q_{new} and q has a high edge possibility, then install q_{new} in T as a child of q

First, the algorithm selects the tree T to expand. Next, a node q is picked from T with highest possibility distribution $\Pi(q)$. Finally, a collision-free configuration q_{new} is picked at distance less than ρ from q and with high possibility of the edge connecting q and q_{new}. This configuration is the new node. Notice that, starting with a neighbourhood of radium ρ, a series of node candidates is selected at random from successively smaller neighbourhoods of q. When a candidate q_{new} which connects with q has high edge possibility tests collision-free, it is retained as the new node. So the fuzzy PRM Planner locally adjusts the sampling resolution as is the case with the original PRM planner in [5].

3.3 Tree Connection

Finding a path from q_i to q_g is to check if the two trees rooted by q_i and q_g, respectively, can be connected. For this purpose, we should have two nodes from the two trees, respectively. Suppose we have a node, say q_{new}, that has been just added to the current tree:

> *which node,* say *q' , should be picked from the other tree to check if the two trees are connected?*

For this, we

> *pick a node q' from the corresponding tree at distance less than ρ from m and with the highest possibility of the edge connecting q and m.*

This is formalized as follows:

Algorithm CONNECT_TREES
Step 1: $q_{new} \leftarrow$ most recently created node
Step 2: Repeat until a node q' has been picked from the tree not containing q_{new}
Step 2.1: Pick a configuration q' at distance less than ρ from q_{new}
Step 2.2: If the possibility of the edge connecting q' and q_{new} is a highest one, then
  Step 2.2.1: Connect m and m' by a bridge ω
  Step 2.2.2: $\tau \leftarrow$ path connecting q_i and q_g
  Step 2.2.3: Return (τ)
Step 3: return *nil*

This connects the two trees by a segment, called *bridge*, joining q_{new} and q' if these two nodes are less than ρ apart and the edge possibility of the bridge is the highest one. That means that either the edge possibility of the bridge is up to 1.0, or the edge possibility of the bridge is less than 1.0 but is highest among all other nodes. Then the bridge creates a path τ joining q_i and q_g in the fuzzy roadmap. The segments along τ, including the bridge, are now ready for collision testing. CONNEC-TREE returns *nil* if there is no any m' at distance less than ρ from m.

3.4 Path Testing

For each edge in path τ, it must be safe, i.e., collision free. In order to test if an edge u is safe, it is impossible to test all points of u. We adopt the same method as in the original PRM [5], which is to uniformly divide u into equally distant subsections. If each subsection is less than the given threshold, namely, it is short enough, and the two endpoints of the subsection have been tested collision-free, u can be considered to be safe.

When a collision is detected, the path testing terminates. Then the colliding edge u is removed from the roadmap. The removal of u disconnects the roadmap into two trees again. The procedures of *tree expansion*, *tree connection* and *path testing* are repeated. It should be noticed that, however, if the colliding edge u is not the bridge that is created in procedure *tree connection* to connect the two trees, the removal of u results in a transfer on nodes from one tree to the other.

4. Related Work

The traditional PRM developed in [6] builds a roadmap of nodes in the configuration space (C-space) of the robot. It has been identified that the traditional PRM planners spend most of their time performing collision checks [5]. Several approaches have hereby adopted to reduce the overall cost of collision checking. Since most local paths in a PRM are not on the final path, delaying collision tests until they are absolutely needed is considered as a promising approach. In [5], adopting some approaches (including delaying collision tests), a new PRM planner has been developed, called SBL, for Single query, Bi-directional, Lazy in collision checking. We apply these techniques in the context of a workplace containing a set of obstacles, a set of dangerzones, and a set of desirezones instead of only a set of obstacles in [5]. Building fuzzy roadmap where each edge is associated with a possibility, our approach is more flexible in path planning.

To handle manipulation planning for a system consisting of a single redundant robot arm manipulating a single movable object with a finite set of stable placements, a two level fuzzy PRM is introduced in [10], in which each

edge of the roadmap is annotated with a probability. Only obstacles are considered in [10] and so a probability associated with an edge only depends on the length of the edge. In our approach, an edge possibility is mainly determined by where its two nodes are located (desirezones, common zones, and dangerzones).

Introducing the notion of *dangerzone* into path planning is first presented in [3], where a dangerzone is an area that should be avoided as much as possible by the robot. Using probabilistic roadmap methods (PRM) developed originally in [6], the paths among obstacles and dangerzones in [3] do not collide with the obstacles and may intersect with the dangerzones. But this should be avoided as much as possible. For this purpose, in [3], the pieces of the paths in which the robot completely penetrates the dangerzones are not accepted. When it happens, such pieces of the paths should be replaced by the new pieces of the paths in which the robot partially lies in the dangerzones and partially lies in the non-obstacle areas. Here the new piece of the paths always touches the boundary of one dander zone. So it is assumed in [3] that the dangerzones do not intersect. Only dangerzones and obstacles are considered and a local planner is used to try to stay out of dangerzones in [3]. The roadmap used in [3] is a classical one. No any possibilities are associated with the edges of the roadmap according to the position relationships among the nodes and the dangerzones.

In [4], obstacle danger level is taken into consideration in path planning. Using fractional potential maps, the paper describes the two optimization methods tested: the A* algorithm and the Fast-Marching technique, and compares and illustrates the efficiency of the two approaches through a vehicle path planning application in a fixed obstacle environment.

5. Conclusion and Future Work

We have just introduced a fuzzy PRM planning method which builds fuzzy roadmaps in which each edge between nodes are associated with possibility degrees that characterize their degree of overlapping with dangerzones and desirezones. That way, the possibility degree associated with an edge in the fuzzy PRM is an estimate of the chance that the edge is actually feasible, i.e., collision free, based on the hard and soft constraints.

It should be noticed that the dengerzones or the desirezones in our current approach have the same weights. We may hope the robot to avoid one dangerzone with the highest priority and another dangerzone with a lower priority. It is not difficult to extend our approach developed in the paper for path planning under such a situation. The only thing we need to do is to consider the effect of the weights when we construct the fuzzy roadmap and calculate the possibility degree associated with an edge of the fuzzy roadmap.

We are currently working on implementing the algorithm developed above and applying it simulation environments.

6. Acknowledgements

This work is supported by a grant from the Canadian Natural Science and Engineering Research Council.

References:

[1] J. C. Latombe, *Robot motion planning*, (Boston, Kluwer Academic Publishers, 1991).

[2] M. H. Overmars, Recent developments in motion planning, *Lecture Notes in Computer Science 2331, Springer Verlag*, 2002, 3-13.

[3] D. Sent & M. H. Overmars, Motion planning in environments with dangerzones, *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, Seoul, Korea, 2001, 1488-1493.

[4] P. Melchior, B. Orsoni, O. Lavialle, A. Poty & A. Oustaloup, Consideration of obstacle danger level in path planning using A* and fast-marching optimisation: comparative study, *Signal Processing*, *83* (11), 2003, 2387-2396.

[5] G. Sanchez & J. C. Latombe, A single-query bi-directional probabilistic roadmap planner with lazy collision checking, *Proceedings of the 9th International Symposium on Robotics Research*, Snowbird, Utah, USA, 2001.

[6] L. E. Kavraki, P. Svestka, J. C. Latombe & M. Overmars, 1996, Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces, *IEEE Transactions on Robotics and Automation*, *12* (4), 1996, 566-580.

[7] R. Bohlin & L. E. Kavraki, Path planning using lazy PRM, *Proceedings of the 2000 IEEE International Conference on Robots and Automation*, San Francisco, California, USA, 2000, 521-528.

[8] L. A. Zadeh, Fuzzy sets as a basis for a theory of possibility, *Fuzzy Sets and Systems*, *1* (1), 1978, 3-28.

[9] L. A. Zadeh, Fuzzy sets, *Information and Control*, *8*, 1965, 338-353.

[10] C. Nielsen & L. Kavraki, A two level fuzzy PRM for manipulation planning, *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Takamatsu, Japan, 2000, 1716-1722.