MEDIAL AXIS LOCAL PLANNER:
LOCAL PLANNING FOR MEDIAL AXIS ROADMAPS

A Thesis

by

KASRA MEHRON MANAVI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2012

Major Subject: Computer Science

MEDIAL AXIS LOCAL PLANNER: LOCAL PLANNING FOR MEDIAL AXIS ROADMAPS

A Thesis

by

KASRA MEHRON MANAVI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Nancy M. Amato
Committee Members,	Suman Chakravorty
	Dezhen Song
Head of Department,	Duncan M. Walker

May 2012

Major Subject: Computer Science

ABSTRACT

Medial Axis Local Planner:

Local Planning for Medial Axis Roadmaps. (May 2012)

Kasra Mehron Manavi, B.S., University of New Mexico

Chair of Advisory Committee: Nancy M. Amato

In motion planning, high clearance paths are favorable due to their increased visibility and reduction of collision risk, such as the safety of problems involving human-robot cooperation. One popular approach to solving motion planning problems is the Probabilistic Roadmap Method (PRM), which generates a graph of the free space of an environment, referred to as a roadmap. In this work we describe a new approach to making high clearance paths when using PRM. The medial axis is useful for this since it represents the set of points with maximal clearance and is well defined in higher dimensions. However, it can only be computed exactly in workspace. Our goal is to generate roadmaps with paths following the medial axis of an environment without explicitly computing the medial axis.

One of the major steps of PRM is local planning, the planning of motion between two nearby nodes. PRMs have been used to build roadmaps that have nodes on the medial axis, but so far there has been no local planner method proposed for connecting these nodes on the medial axis. These types of high clearance motions are desirable and needed in many robotics applications. This work proposes Medial Axis Local Planner (MALP), a local planner which attempts to connect medial axis configurations via the medial axis. The recursive method takes a simple path between two medial axis configurations and attempts to deform the path to fit the medial axis. This deformation creates paths with high clearance and visibility properties. We have implemented this local planner and have tested it in 2D and 3D rigid body and 8D and 16D fixed base articulated linkage environments. We compare MALP with a straight-line local planner (SL), a typical local planner used in motion planning that interpolated along a line in the planning space. Our results indicate that MALP generated higher clearance paths than SL local planning. As a result, MALP found more connections and generated fewer connected components as compared to connecting the same nodes using SL connections. Using MALP connects nodes on the medial axis, increasing the overall clearance of the roadmap generated.

To My Family

ACKNOWLEDGMENTS

This work could not have been done without the help and support of a great set of individuals. They have all played important parts in both this work and my life.

First and foremost I would like to thank my advisor Dr. Nancy M. Amato for all of her support and guidance. I learned a lot from you and you showed me how much I can achieve if I push myself. Texas was quite an adventure for me and I am thankful for having you help me through it.

I would like to thank my committee members Dr. Suman Chakravorty and Dr. Dezhen Song for their cooperation, patience and support. I am truly grateful.

I would like to thank members of the Algorithms and Applications group in the Parasol Lab. Dr. Shawna Thomas was a huge help in both mentoring me and inspiring me to work hard, her aid was invaluable. I would also like to thank my collaborators Jory Denny, Sam Jacobs and Aditya Mahadevan. You guys have made work interesting, fun and exciting and enjoyed my time working with you all.

Thank you family for all your support, and a personal thank you to Pavan Dharwadkar and Adam Fidel. You guys helped me more than you know.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

One of the major problems in robotics is that of finding a valid, collision-free path for a robot through a given environment. Motion planning [1] has been extensivly studied and has applications in a variety of domains, from robotic path planning to virtual prototyping/virtual reality [2] to computational biology [3]. Thus it is important to find higher quality and more accurate methods for solving the motion planning problem.

Sampling-based planners [4] were a major breakthrough in motion planning. These methods were able to solve previously unsolvable problems, including high-dimensional problems. Sampling-based planners have been shown to be probabilistically complete, meaning that the probablity of finding a solution approaches 1 as the sampling density is increased. However narrow passages, or tightly constrained environments, still remain difficult to traverse.

There have been many variants to the original algorithm that address the weakness of sampling-based planners in these narrow regions by producing better samples [5] [6] [7] [8]. One method, Medial Axis PRM (MAPRM) [9], increases samples in the narrow passage by retracting all configurations, valid or not, to the medial axis of the free space. The medial axis is defined as the set of points of a space having more than one closest point on the space boundary. The medial axis definition extends into higher dimensions and can be viewed simply as the set of configurations with maximal clearance. MAPRM improves the quality of the samples generated and has been shown to be more effective than uniform random sampling in narrow corridors. This method increases the number of nodes found in narrow corridors in a way that is independent of the volume of a corridor, depending solely on the volume of the obstacles surrounding

The journal model is *IEEE Transactions on Automatic Control.*

it. MAPRM produces samples with high clearance, thus high visibility, and connections have a better chance at success. Thus, this increase in narrow passage sampling allowed for solutions to be found in quicker time using fewer, but higher quality samples.

When these medial axis samples are connected using simple local planners (e.g., SL), the roadmap generated contains connections which diverge from the medial axis. Hence the resulting roadmap has paths that may have sub-optimal clearance for planning. Also, depending on how well sampling covers an environment, simple connection strategies might not be enough and increased sampling potentially leads to over sampling and longer running times.

This paper introduces the Medial Axis Local Planner (MALP), a local planner intended to produce high clearance connections. The local planner attempts to deform a simple path between two medial axis configurations to the medial axis. MALP begins by pushing the middle configuration of a path between start and goal configurations to the medial axis. MALP then recurses on each new path, the segment between the newly pushed configuration and an existing configuration until it has converged to a solution close enough to the medial axis or until the maximum number of iterations is reached. A maximum number of recursions is a user specified variable since a connection may not exist since the medial axis may be disjoint, samples may be too far away, or it may be a degenerate connection. MALP can be used with MAPRM to build roadmaps that lie entirely on the medial axis. MAPRM and MALP can utilize approximate methods of medial axis computation in cases where explicit medial axis computation would be prohibitively expensive if not impossible due to the nature of the problem studied [10].

We test MALP in 2D and 3D rigid body and 8D and 16D fixed-base articulated linkage environments. MALP is tested against SL local planner, a local planner typically used in motion planning. Our results show that MALP generates paths with higher clearance than SL, leading to higher connection success rates and resulting in larger connected components. Though MALP may be more costly, it can be effective at planning paths between narrow passage configurations and complex regions of C_{space}.

This paper's contributions include the following:

- MALP, a local planner which attempts to connect two medial axis nodes via the medial axis without explicitly computing the medial axis.
- A foundation for roadmap construction along the medial axis in arbitrary dimensions.

Our results show that MALP has greater success rates of connection than SL in all environments tested. As we expected, the number of edges MALP found increased as the number of maximum iterations increases. This led to a reduction of the number of connected components in the roadmap.

This thesis describes MALP, our proposed medial axis local planner. We describe in Chapter II the related work to our method. This includes medial axis motion planning, local planning and path deformation. In Chapter III we describe MALP and improvements made to approximate medial axis retraction. Chapter IV describes the experiments run and thier results. Finally, in Chapter V we discuss our results and conclusions about MALP.

CHAPTER II

RELATED WORK

In this section we discuss the related work. We first give an introduction to sampling-based motion planning and Probabilistic Roadmap Methods (PRM). We then provide an introduction to medial axis PRMs which utilize high clearance configurations. Finally, we present an overview of local planning and path deformation.

1. Sampling-Based Motion Planning

A robot is a moveable object whose position and orientation can be defined by d parameters, or degrees of freedom (DOFs). These parameters define the robot placement, or configuration, in an environment. These d parameters can be used to describe the robot as a point in an d-dimensional space. This space is referred to as configuration space, or C_{space} and includes all possible configurations, valid and invalid [11]. All valid, or feasible, configurations are considered to be in the subset C_{free} and all invalid, or infeasible, configurations are in C_{obst}. The motion planning problem has now become a problem of finding a valid series of configurations in C_{free} between a start and a goal. Sampling-based motion planners attempt to explore C_{space} by sampling and connecting configurations in C_{free}.

One important sampling-based planner is the Probabilistic Roadmap Method (PRM) [4]. PRMs build a roadmap, a graph that represents the connectivity of C_{free}. This graph can then be used as a foundation for traversing C_{free} The PRM algorihtm is outlined in Algorithm 1.

As can be seen in Figure 1, the planner begins the construction phase with node generation. Random samples are generated and valid configurations in C_{free} are saved and added to the roadmap as nodes. The second phase of construction is connection

Algorithm 1 Probablistic Roadmap Method

```
INPUT: Environment $e$, number of configurations $n$, nearest neighbors $k$,
        distance metric $dm$, local planner $lp$, and $start$ and $goal$ configurations
OUTPUT: Roadmap $r$, with $n$ configs and path $p$ if it exists
  GenerateNodes( $e$, $r$, $n$)
  ConnectNodes( $e$, $r$, $k$, $dm$, $lp$)
  ConnectQuery( $start$, $goal$, $e$, $r$, $k$, $dm$, $lp$)
  FindPath( $start$, $goal$, $p$ )
```

where *distance metrics* are used to determine the nearest neighbors in the roadmap, in the case of Figure 1, k=2, k being the number of nearest neighbors. A *local planner* is used to determine if a connection can be made between neighbors. Successful connections are added to the roadmap as edges between nodes. Once the roadmap is constructed, we can attempt a query to find a planned solution. Queries are processed first by connecting start and goal configurations to the roadmap. From there a pathway is extracted if one exists using simple shortest path graph search algorithms.

2. Medial Axis Motion Planning

The Medial Axis Probabilistic Roadmap Method (MAPRM) is a variant of PRM which utilizes the medial axis [9] [12] [10]. This algorithm is outlined in Algorithm 2. Random configurations are sampled and then retracted to the medial axis, i.e., pushed to areas of C_{free} with higher clerance. as can be seen in Figure 2. This is done by finding the configuration with minimal clearance/penetration distance from an initial configuration and using it to determine a direction to retract the initial configuration to the medial axis. A retracted configuration is pushed out of C_{obst} if necessary, then pushed away from C_{obst} til a second configuration in C_{obst} is found to be equidistant to the first.

Clearance is an important computation and can be performed in both exact and

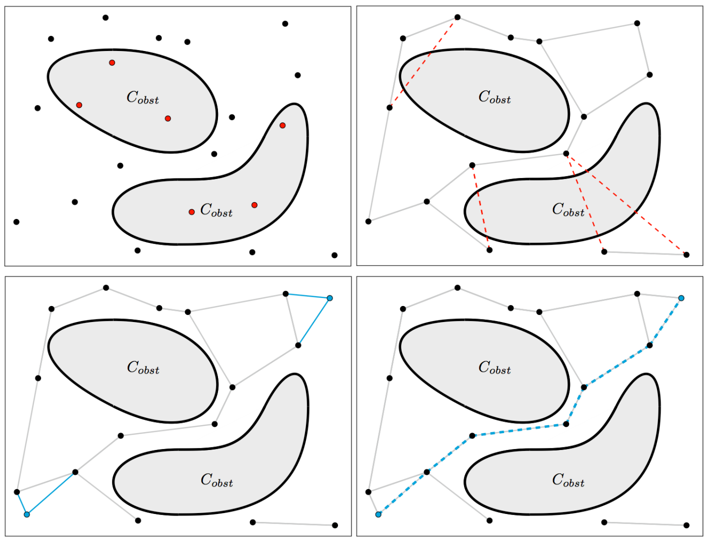


Fig. 1. PRM is performed by generating random configurations and saving the valid ones (*top left*), then finding the k nearest neighbors (here k=2) of all the configurations and connecting them (*top right*). Start and goal configurations are add to the graph (*bottom left*) and the shortest path solution is determined (*bottom right*)

approximate fashions. Configurations on the medial axis have high visibility and are easier to connect. It should be noted that MAPRM only samples on the medial axis and nodes are connected by local planners which generally do not make connections on the medial axis.

There are other medial axis planners that have been developed but are restricted to workspace. A Framework for Using Workspace Medial Axis in PRM [13] explores the workspace medial axis, not the C_{free} medial axis, by computing a polygonal approximation of the workspace medial axis. Another method is the Voronoi Based Framework for Motion Planning [14] which pre-computes the Generalized Voronoi Diagram (GVD) of the workspace using graphics hardware and uses that as a foundation for environment traversal. This method uses randomized path planning to traverse invalid segments of a robot moving along the GVD pathway. Another method uses sensor based exploration to incrementally construct a heirarchical generalized Voronoi graph (HGVG) [15] [16].

Algorithm 2 MAPRM Sampling

```
INPUT: Environment $e$, number of configs $n$
OUTPUT: Roadmap $r$, with $n$ Medial Axis configs
for $i$ from 1 to $n$ do
    Config $a = e$.GetRandomConfig()
    if !IsValid( $a$, $e$ ) then
        $a$.PushOutOfCollision()
    end if
    $a$.PushToMedialAxis()
    $r$.AddConfig( $a$ )
end for
```

This method attempts to map an environment using a series of sensors to provide data which is used numerically construct the HGVG and was tested on physical systems as well as in simulation. These methods work well in workspace but are not general enough to plan on the medial axis in arbitrary dimensions besides the HGVG.

3. Local Planning

A local planner is the check performed to see if two configurations are connectable and is usually run on the nearest neighboring samples in C_{space}. Local planners are intended to be inexpensive in terms of computation since they are performed many times. SL local planning linearly interpolates a series of configurations that transition from one configuration to the other in C_{space}. Another popular local planner is rotate-at-s [17] which translates a configuration to a specified percentage of its pathway towards its goal, rotates it, then completes the translation to the goal configuration. This helps with obstacle-based planners such as [5] and [6]. Both SL and rotate-at-s are quick and simple local planners but can lead to many failures in sparsely sampled areas of

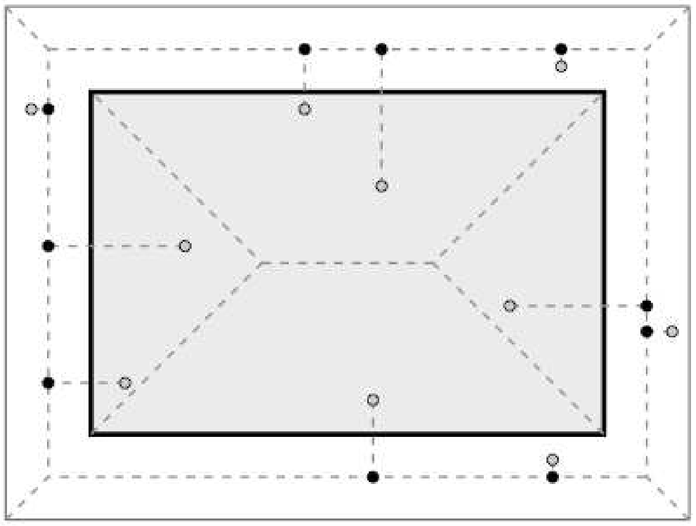


Fig. 2. MAPRM sampling is performed by generating random configurations and retracting them to the medial axis. Here, the initial configurations are colored gray and the retracted configurations are colored black.

the C_{space} or more complicated areas of an environment.

More expensive local planners such as those based on A* [18] and Path Planning in Expansive Spaces [19], are better at finding connections. A* uses a best-first search over a resolution sized grid in C_{space} and finds the lowest costing path. Path Planning in Expansive Spaces grows trees rooted at the nodes and connects them once their visibility regions become overlapping. These local planners when used between further away nodes potentially require large amounts of storage along the edge, requiring many samples in the region of connection. These algorithms are expensive and generally have a timeout parameter to ensure problem stoppage in a reasonable time.

In [20], a simplified potential field local planner was used to analyze the reachability of sampling based planners. This local planner attempts a SL connection and either reaches the goal or if while stepping out, collides with an obstacle. If a collision happens, a series of random directions oriented towards the goal are tested for collision. The best candidate (free and closest to the goal) is used to step out and avoid the collision. Stepping towards the goal continues till either the goal is found or finding a direction to avoid collision fails. One of the major conclusions made from this study was that for motion planning, especially narrow passage problems, the major problem was not covering C_{free}, but was instead achieving good connectivity. Strategies they recommend to resolve this include using hybrid strategies in difficult areas of C_{space} and by employing more powerful local planners to connect nodes.

4. Path Deformation

Elastic Bands [21] and Elastic Straps [22] address real-time obstacle avoidance in a dynamic environment. Elastic Bands/Straps start with an initially collision free path and incrementally modify the path to maintain a smooth, collsion free path. This method does not utilize any explicit C_{space} information, it relies on *protective bubbles* defined in workspace to maintain C_{free} information, and has the constraint of requiring an initial collsion free path. Methods used in [23] create high-quality paths by refining a path in terms of length and clearance. These methods require an initial valid path from which each intermediate node is retracted to the medial axis to increase clearance. The path is then pruned to reduce the overall length.

Path Deformation Roadmaps [24] rely on the notion of path deformability indicating whether or not a specified path can be continuously deformed into another existing path. This method only looks at homotopy classes and is dependent on visibility to determine if a deformation is possible. The Reachability Roadmap Method (RRM) [25], intended for 2D and 3D virtual environments, takes an initial roadmap and query solution and adds "useful" nodes and edges to the roadmap to improve the solution. If a potential node reduces the distance of the existing shortest path connecting its nearest neighbors, it is considered "useful" and added to the roadmap. The roadmap is then reconnected by rearranging and adding edges to better fit the new nodes, and then retracting these edges to the medial axis. RRM showed that alternative and reasonably short query paths can be found by enhancing an existing roadmap.

Reactive Robot Motion using Path Replanning and Deformation [26] uses path deformation in online path replanning by pushing invalid parts of a SL path away from obstacles. The midpoint of an invalid pathway is pushed to the outside edge of the obstacles repulsion area, an area defined as too close to an obstcle. The two new SL paths formed are recursed upon until a valid path is found or the replanner times out.

CHAPTER III

MEDIAL AXIS LOCAL PLANNING (MALP)

We begin this chapter introducing MALP, a local planner which produces paths along the medial axis. We then discuss approximate C_{space} clearance and how it impacts medial axis retraction. Finally, we introduce a heuristic used to increase the accuracy of medial axis retraction using approximate clearance computation.

1. MALP

MAPRM generates samples on the medial axis, but connecting them may reduce the overall clearance properties of a roadmap since edges do not lie on the medial axis. Medial Axis Local Planner (MALP) computes connections that reside on the medial axis. This is done by deforming a path between two medial axis configurations to be ϵ-close to the medial axis. The path is considered ϵ-close to the medial axis if all the configurations along the path are no greater than ϵ away from the medial axis. Algorithm 3 outlines the approach.

Algorithm 3 begins by taking two medial axis configurations and determining if a path between them is ϵ-close to the medial axis. This path is tested for both collisions and for ϵ-closeness to the the medial axis. To test for ϵ-closeness, intermediate configurations along the path are retracted to the medial axis and their displacement is measured. This is the same retraction procedure as used in MAPRM sampling [9]. If the path is not ϵ-close, the middle configuration of the path is retracted to the medial axis. The two new paths generated between the retracted midpoint and endpoints are now recursed upon. MALP continues to recurse until it has converged to a solution that is considered ϵ-close to the medial axis or reaches an exit case. If the maximum number of iterations has been reached or the configurations being connected are closer

Algorithm 3 MALP

```
INPUT: Medial axis configurations $a$ and $b$, distance $\epsilon$, and iteration $itr$
OUTPUT: A path $\epsilon$-close to the medial axis if feasible, else $\emptyset$
OTHER: A local planner $lp$, validity checker $vc$ and maximum iteration $itr_{max}$
// Let vc be a function that returns true if configurations are collision free and
$\epsilon$-close
if ( $itr \leq itr_{max}$ ) then
   Return $\emptyset$
end if
$P_0$ = $lp$.GetValidPath( $a$, $b$, $vc$ )  // Returns valid path $P_0$ based on the validity
definition
                                            // given by vc, or $\emptyset$ if the path doesn't exist
if ( $P_0$ != $\emptyset$ ) then
   Return $P_0$
end if
$mid$ = PushToMedialAxis( ( $a + b$ ) / 2 )
$P_1$ = MALP( $a$, $mid$, $\epsilon$, $itr$+1 )
if ( $P_1 == \emptyset$ ) then
   Return $\emptyset$
end if
$P_2$ = MALP( $mid$, $b$, $\epsilon$, $itr$+1 )
if ( $P_2 == \emptyset$ ) then
   Return $\emptyset$
end if
Return ( $P_1 \circ P_2$ )
```

than the environment resolution and the path has still not converged, the attempt is considered a failure. This new edge, if successfully generated, connects medial axis nodes along the medial axis. A step by step example is described in Figure 3.

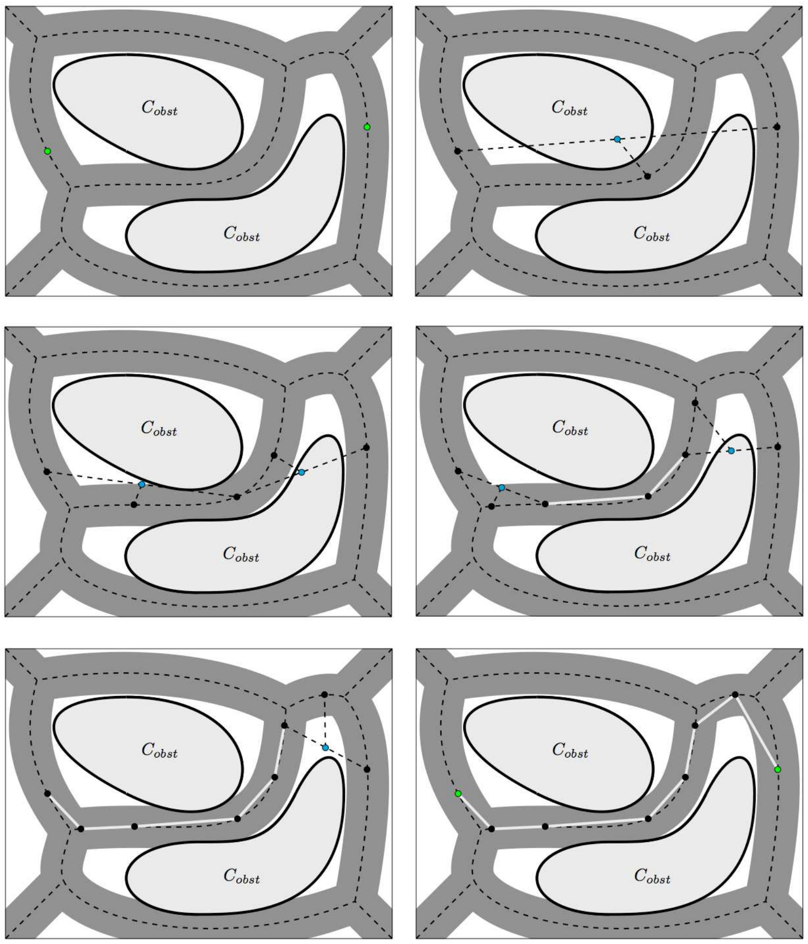


Fig. 3. Using MALP to connect two MAPRM configurations with SL as the base local planner. MALP computes the SL path between two medial axis configurations, retracts the midpoint to the medial axis and recurses on these new SL paths as necessary. The dashed lines show the medial axis and the gray lregion is the ϵ-close area surrounding it.

If ϵ is very small MALP can be very expensive. Paths produced using different ϵ values have different clearance properties. Larger ϵ valued paths require fewer calls to push the medial axis and solve in fewer iterations since paths have more leeway in thier traversal of the medial axis.

A connection may not exist since the medial axis may be disjoint, samples are too far away, or a degenerate case is encountered. For example, the local planner can get caught up when two medial axis configurations are on opposite sides of a symmetric obstacle as can be seen in Figure 4. This degenerate case is a local minima, the retracted middle configuration ends up exactly where an outer configuration lies. This cycle will continue since the new midpoint is in the same location as the last. Thus, we use a maximum number of iterations as an input parameter to stop computation of these non-connectable paths.

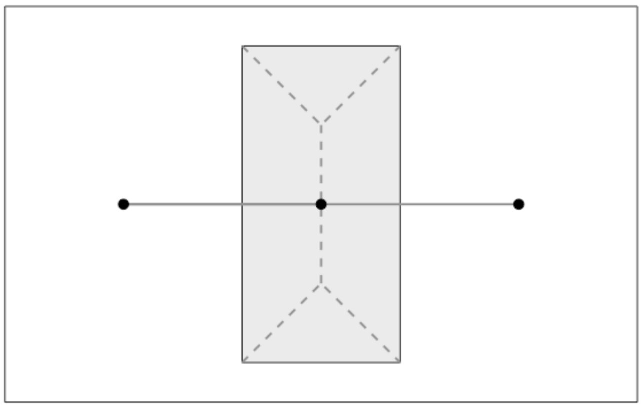


Fig. 4. An example of a degenerate case using MALP. Two medial axis configurations on opposite sides of a symmetric obstacle. The middle configuration ends up being retracted to one of the configurations being connected, resulting in a new middle configuration at the same position as the previous one.

Using a medial axis local planner, although potentially more expensive, provides a higher clearance path between two medial axis configurations on the medial axis. We now discuss two details of the approach that deal with medial axis retraction, approximate C_{space} clearance computations and history retaining.

2. Approximate C-Space Clearance

The clearance computation is an important operation in medial axis motion planning. Clearance computations can be performed in both exact and approximate fashions. Exact clearance works well for workspace planning but does not generalize to C_{space}. 2D and 3D environments can utilize polygonal information to compute exact clearances and it works well for rigid bodies. Exact clearance computations cannot, however, be guarenteed using non-convex obstacles, particularly when computing penetration. Take for instance decomposing a 'T' shaped obstacle into convex obstacles into two boxes, top and bottom. An internal face is introduced at their intersection that can be used to calculate incorrect clearance/penetration distances, eventually leading to erroneous medial axis retraction. Exact clearance computations also cannot take into account rotational and internal degrees of freedom, i.e., links of an articulated robot. Approximate C_{space} clearance becomes important in higher DOF problems where workspace clearance is not sufficient for planning. Approximate C_{space} clearance is better for these higher DOF problems since the workspace obstacle clearance becomes less of a factor and self-collisions can be factored in. To approximate the C_{space} clearance, a series of random rays are shot out from the configuration and stop on the boundary of C_{free} [10]. An example can be seen in Figure 5. The number of random rays is an input parameter to MAPRM and MALP. As the number of rays increase, accuracy increases but so does computation time.

3. History Heuristic

We present a heuristic called history which attempts to improve the quality of approximate medial axis retraction from [10]. As can be seen in Figure 6, when retracting to the medial axis with approximate configuration clearance, the only reference for finding the medial axis is the clearance distance. The witness points (cyan) are approximate

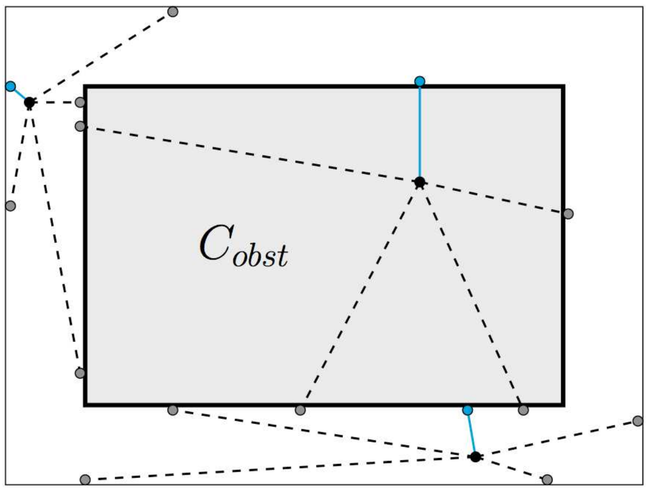


Fig. 5. Approximating C_{space} clearance is done by shooting out random rays and using the distance of the shortest ray as minimum clearance.

and can't be used as a reference since each step of the retraction computes a new approximate clearance, resulting with a different witness.

As a configuration is being retracted onto the medial axis, the approximate clearance values are added to a queue. This queue is used essentially as a lo-fi filter, looking for a peaking trend in the clearance values. This helps approximations with lower ray counts by trying to compensate for the noisy clearance data collected. When retracting onto the medial axis we would expect to see a unimodal sequence, an increase in clearance till the medial axis was reached, then a decrease. Notice in Figure 6 the third configuration of the retraction has over approximated its clearance, resulting in a value greater than the next retraction steps clearance. This can be seen as a false positive if we adhere to a strict peaking trend policy. As a configuration is being retracted, the clearance values are added to the history queue, up to a specified size, and retains the most current clearance values.

The history queue is then iterated over and a ratio of positive to negative derivative values is calculated. Once this ratio becomes becomes 50/50, we can assume the medial axis lies within the history list. Figure 7 shows examples using different sized history

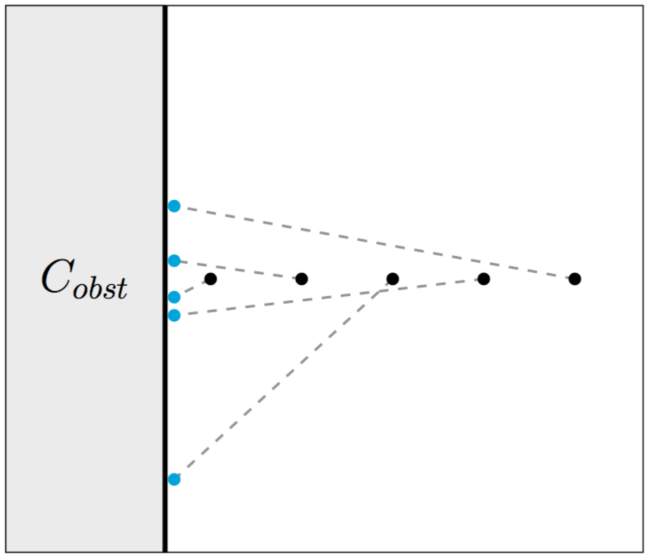


Fig. 6. The retraction process is performed by retracting in the opposite direction of minimal clearance till maximal clearance or a different witness point is found

lengths (l) to determine the span surrounding the peak. We can see that the longer history queue spans signified by pairs of squares (l=17) and circles (l=9) contain the real peak. Shorter history lenghts can end finding false positive peaks early in the retraction, such as the spans signifed by pairs of '+' (l=5) and '*' (l=3).

The span of the history queue is then used as the foundation for a modified binomial-type search to find the peak. This binomial search places 5 equally spaced points spanning the history line segment. These points define 4 line segments that have their derivative computed and used to search for a peaking trend. The 2 consecutive segments which best describe a peak, a positive followed by a negative and whose shared point has the highest clearance, or the highest end of a monotonic set of clearance values is then recursed upon. The recursion stops when the distance between the two points defining the span reaches a specified Δ.

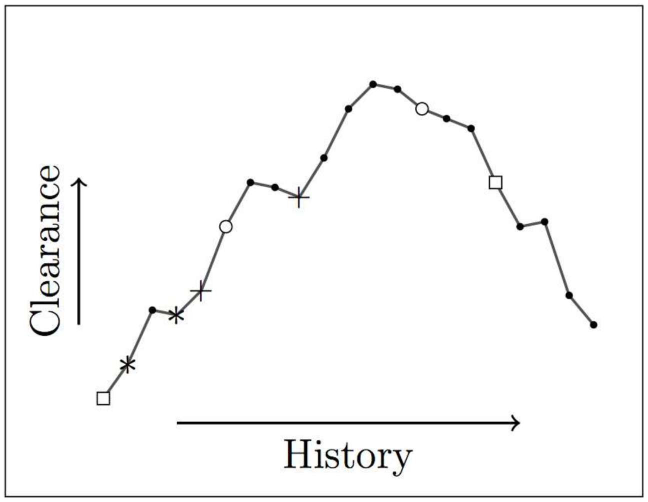


Fig. 7. History spans using different history lenghts and the peaks they determine. Spans signified by pairs of squares (l=17) and circles (l=9) contain the real peak, but spans signifed by pairs of '+' (l=5) and '*' (l=3) determine a false positive peak prematurely.

CHAPTER IV

EXPERIMENTS

In our experiments we study several different aspects of MALP and medial axis path planning: MALP performance compared to SL, MALP roadmap and path clearance properties compared to SL, quality of the history heuristic, and using MALP with approximate retraction. Figure 8 visualizes the differences between a map generetated using SL (*right*) and MALP (*left*) in a simple 2D environment.

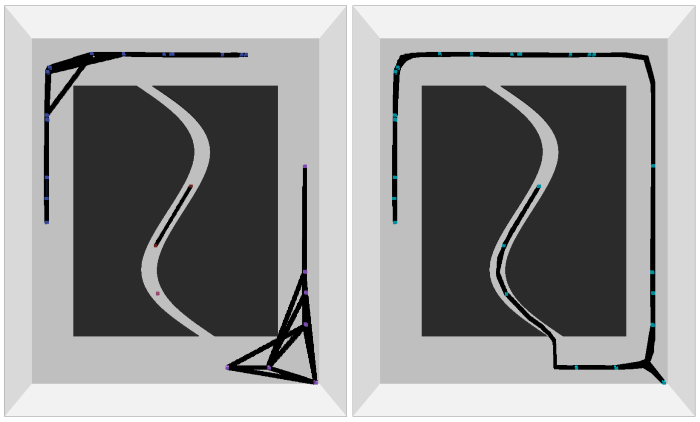


Fig. 8. Roadmaps generated using 25 MAPRM nodes in a simple 2D environment are connected using SL (*right*) and MALP (*left*). Note that more connections are found using MALP along with increased path clearance.

First, we analyze the performance of MALP and compare it to SL local planning. We then compare the paths and roadmaps produced by MALP and SL local planners and analyze their clearance and length. From there we move on to the history heuristic and show that it is effective in producing better approximate medial axis retractions. Finally, we look at MALP using approximate medial axis retraction in higher dimensions.

1. Experimental Setup

Here we discuss the different aspects of the experimental setup. First we describe the environments used and why they were chosen, they can be seen in Figure 9. We then discuss the experiments run and the resulting statistics collected. Finally we go over the implmentation details of the experiment.

a. Environments

We use 2D and 3D environments (Figures 9(a) and 9(b)) to test the performance of MALP. The 2D maze environment (Figure 9(a)) has two solutions, generally uniform clearance values since the hallways throughout the maze are the same width, and is traversed by a point robot. In this environment, connections are difficult since nearest neighbors can be close according to Euclidean distance but far apart when traversing C_{free}. The 3D cluttered environment (Figure 9(b)) has a small box robot and contains 48 thick plate obstacles randomly placed throughout the space. This environment is intended to capture a more heterogeneous C_{space} including free and cluttered spaces as well as narrow corridors.

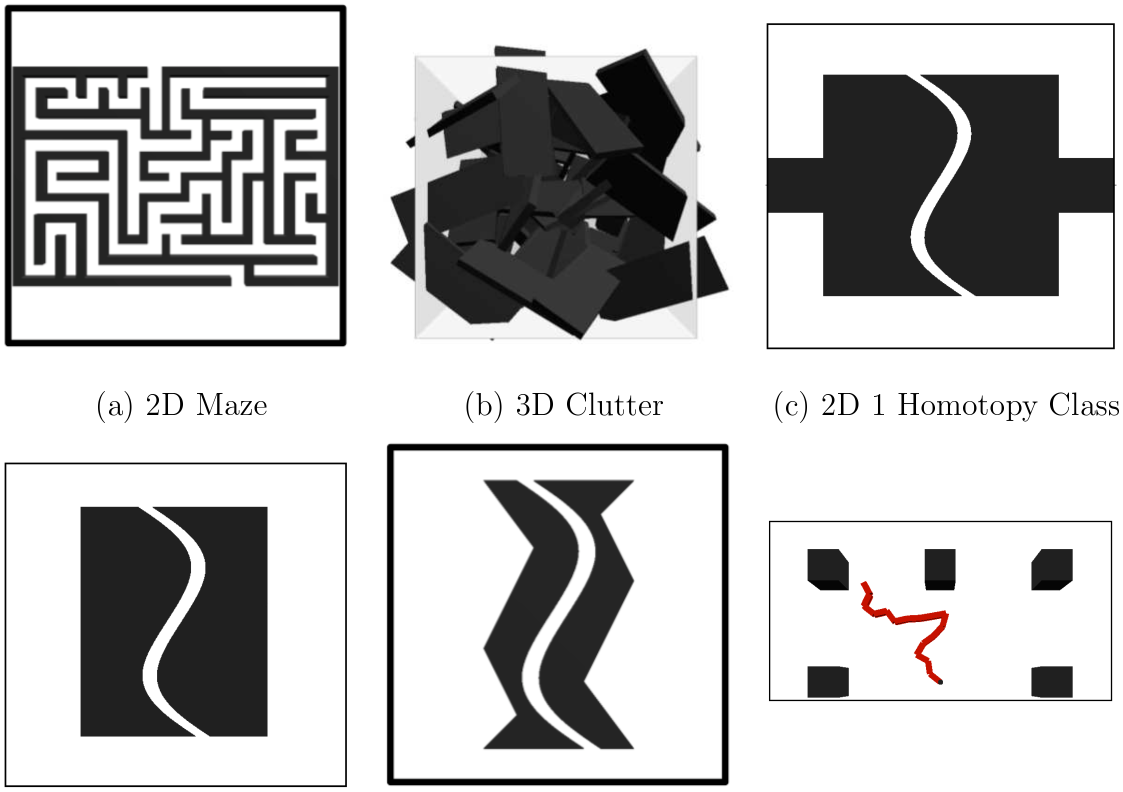
To explore the differences in roadmaps and paths generated using either MALP or SL, we use environments with different homotopic groups. We look at single homotopy class environments in both 2D (Figure 9(c)) and 8D examples. The 2D single homotopy class environment has a single narrow passage and uses a point robot. The 8D grid environment has a set of 8 articulated links with a fixed base on a plane. The environment has a series of compartments that are divided by a set of obstacles. The 8D grid environment is the same as seen in Figure 9(f), but uses 8 articulated links instead of the displayed 16 links and has the same overall robot dimensions. Both of these environments have a query solution and all solution paths can be deformed to all others, thus we have single homotopy class environments. We also look at environments with multiple homotopic classes in 2D (Figure 9(d)) and 3D (Figure 9(b)). The 2D environment contains a narrow passage similar to that in the simple homotopic class environment in Figure 9(c) and uses the same point robot. However, this environment has the outer portion of the obstacles removed, allowing for the definition of 3 homotopic groups. The 3D clutter environment (Figure 9(b)) has a great deal more query solutions which are not deformable to one another due to the complexity of C_{obst}.

To test the performance of the history heuristic, we use a simple 2D environment as seen in Figure 9(e). The 2D simple environment contains two non-convex obstacles defining a narrow passage and is sampled by a point robot.

Finally, we look at 3D and high DOF articulated linkage environments to explore approximate medial axis retraction. 3D clutter is used along with the grid environment (Figure9(f)) using 8D and 16D articulated linkages (16 Link being shown).

b. Experiments

To test MALP performance, we generate a specified number of medial axis samples (100 and 300 in 2D, 200 in 3D) and attempt connections between the five nearest neighbors to each sample as defined by Euclidean distance. Thus, the same edges are attempted by each local planner studied, in our case SL and MALP. Using the different number of samples allowes us to see the performance of different roadmap densities in the case of 2D maze. We examine the effect of epsilon distance and the maximum number of iterations on the construction statistics and properties of the roadmap using MALP. We look at successful connection attempts and the size of the largest connected component as a measure of quality and connectivity, and the number of collision detection calls as a measure of time. We also test the generality of MALP by using different base local planers. We attempt MALP connections using SL and rotate-at-s as the base local planner.



(a) 2D Maze (b) 3D Clutter (c) 2D 1 Homotopy Class

(d) 2D 3 Homotopy Class (e) 2D Simple (f) 8D/16D Grid

Fig. 9. Environments used in our studies include a 2D maze environment, a 3D cluttered environment, 2 2D environments with different homotopic groups, a simple 2D non-convex narrow passage and fixed based articulated linkage environment using 8D and 16D robots (16D shown).

For experiments involving path and roadmap clearance, we generate a set of medial axis samples and connect them using SL and MALP. The newly connected maps are then queried and a solution path is produced. From there, both query solution and roadmap clearances are analyzed. When analyzing paths, we look at minumin, maximum, and average clearances of the configurations along the path and the length of the path. When analyzing roadmaps, we first look at roadmap length and average, minimum, and maximum clearaces followed by average edge minumims and maximums.

For the history heuristic experiment, we sample 100 configurations using different ray count and history length parameters and calculate their distance from the medial axis. We report the average clearance values over all configurations sampled.

For the approximate retraction experiments, different ray counts and epsilon values are studied. The same parameters are analyzed as in the prior MALP performance experiment.

c. Implementation Details

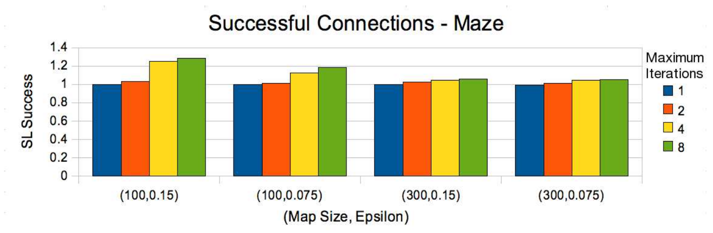
The algorithm was implemeneted and tested in the Parasol Motion Planning Library (PMPL) framework and results averaged over a series of 10 runs. We used the collision detecion library PQP [27] for our experiments. The developement and experimentation were done in Linux environments and compiled using GCC 4.1.2.

2. MALP Performance

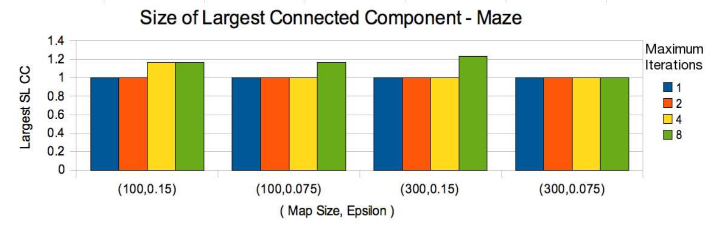
a. MALP for 2 Degrees of Freedom

Figure 10 compares the performance of MALP and SL in the 2D maze environment (Figure 9(a)) using exact clearance computation and the same set of medial axis samples. Roadmap size (i.e,. sampling density), ϵ and the number of MALP iterations were varied. To analyze MALP, we normalize all statistics over SL local planner results to see the performance difference.

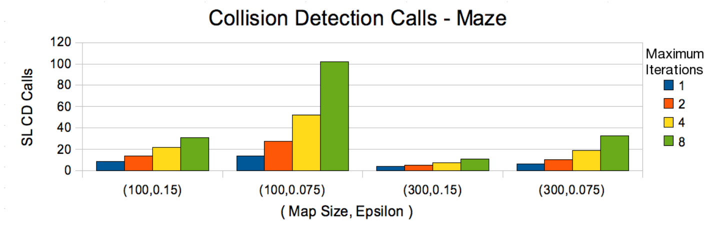
In Figure 10(a), we are looking at the success rate of MALP over SL (*y axis*) for different map sizes, epsilon values and maximum iteration values (*x axis*) using MALP. We see that MALP has equivalent, if not greater, success rates than SL for all roadmap sizes sparse (n=100) and dense (n=300). We see that MALP performs as good if not better than SL in all 2D maze cases at all ϵ values and iteration counts. As expected, the number of edges MALP finds increases as the number of iterations increases. We can see that the sparse roadmap sees a greater benefit using MALP



(a) Successful Connection Attempts



(b) Largest Connected Component Sizes



(c) Collision Detection Call Counts

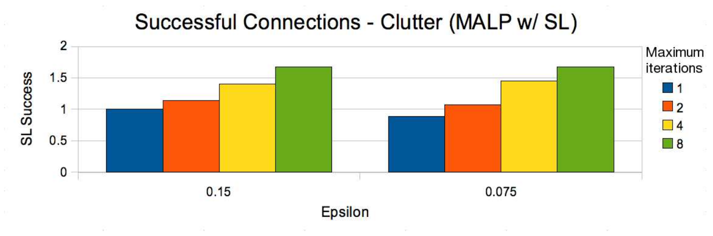
Fig. 10. Results from the 2D maze environment (Figure 9(a)) using MALP with 1, 2, 4 and 8 maximum iterations and ϵ values of 0.15 and 0.075 on both sparse (n=100) and dense (n=300) roadmaps, n being the number of samples in the roadmap. All results are normalized on SL local planner performance.

than a dense roadmap. This is due to the fact that more dense maps have more simple connections, reducing the percentage of difficult connections MALP could make that SL could not. Figure 10(b) plots the relative size of the largest connected component found compared to that of one found using SL local planner. These are plotted against the same map sizes and epsilons values as the prior connections experiment. We see that roadmaps produced using MALP can increase the size of the largest connected component, showing that it can make more unique connections and can better capture the connectivity of C_{free}. In Figure 10(c), we plot the cost of MALP relative to SL local planner (*y axis*) versus the same map sizes and epsilon values as before (*x axis*). We see that MALP is more expensive that SL, the cost being a function of map size, ϵ, and maximum number of iterations allowed. More dense maps have more simple connections for MALP to make, thus reducing the overhead of retraction. Higher ϵ values reduce the cost by relaxing the clearance constraint, but maximum number of iterations looks to be the main factor.

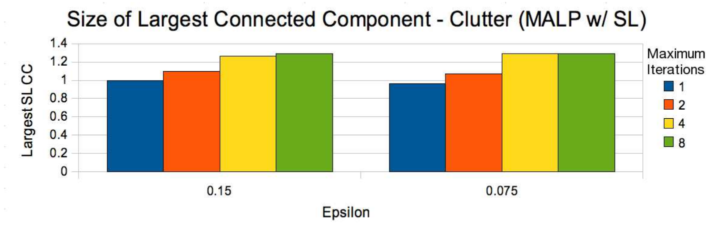
b. MALP for 3 Degrees of Freedom

Figure 11 shows the results for MALP using SL as a base local planner in the 3D cluttered environment (Figure 9(b)) using exact clearance computations while varying ϵ and the number of maximum iterations. Results for the MALP are reported normalized against SL performance.

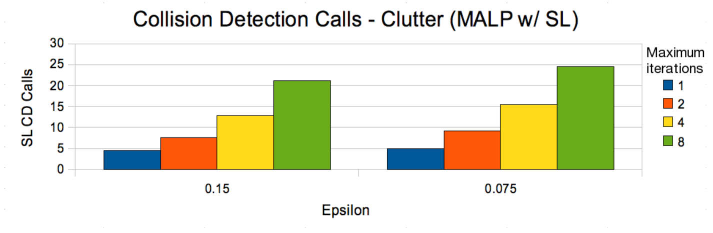
Figure 11(a) shows the success rate of MALP relative to SL versus MALP with different epsilon values and maximum number of iterations. As we saw earlier the 2D maze results (Figure 10(a)), MALP success rate increases with increasing iterations in the 3D case. Note that at 1 maximum iteration using the smaller epsilong value, MALP has a lower success rate than SL. We attribute this initial performance decrease to the value of ϵ being used. A smaller ϵ can reduce the number of connections because even if a free path is available, it is considered a failure if it is not ϵ-close. We again see similar trends in the size of the largest connected component as we saw in 2d maze. Figure 11(b) shows that the size of the largest connected component increases with increasing iteration counts and increasing ϵ. In this instance, setting the maximum iterations to 8, the resulting map was fully connected, all samples were apart of the same connected component. In Figure 11(c), we compare the number of collision detection calls between MALP and SL. Retraction is an expensive operation, but there



(a) Successful Connection Attempts



(b) Size of the Largest Connected Component



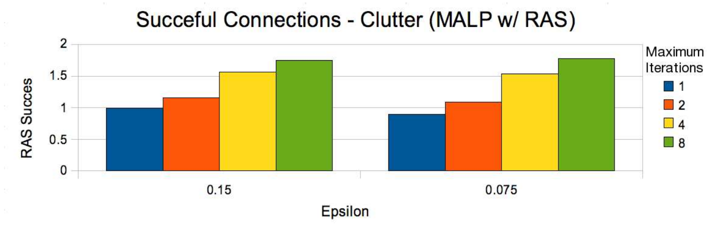
(c) Number of Collision Detection Calls

Fig. 11. Results from the 3D clutter environment (Figure 9(b)) using MALP with SL as a base local planner, values of 1, 2, 4 and 8 for maximum iterations, ϵ values of 0.15 and 0.075 and exact clearance computation.

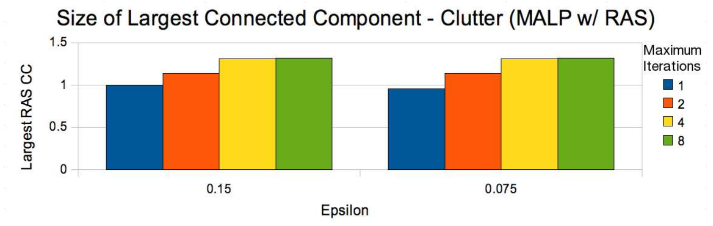
is a guaranteed quality to the paths generated and in many cases a higher success rates and more roadmap connectivity.

To test the generality of MALP, we use MALP to connect a roadmap but using rotate-at-s as the base local planner. Figure 12 shows the results for MALP using rotate-at-s in the 3D cluttered environment (Figure 9(b)) using exact clearance computations while varying ϵ and the maximum number of iterations. Results for MALP are reported normalized against rotate-at-s performance.

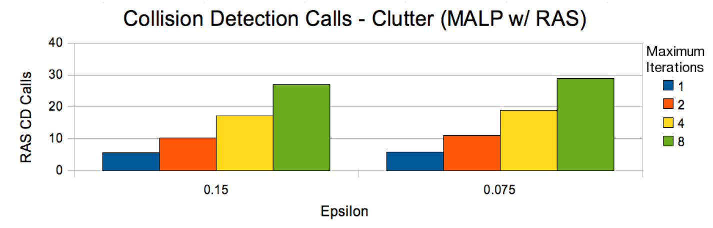
Figure 12(a) plots the success of MALP connections relative to rotate-at-s versus



(a) Successful Connection Attempts



(b) Size of the Largest Connected Component



(c) Number of Collision Detection Calls

Fig. 12. Results from the 3D clutter environment (Figure 9(b)) using MALP with rotate-at-s as a base planner, using values 1, 2, 4 and 8 for maximum iterations, ϵ values of 0.15 and 0.075 and exact clearance computations.

different ϵ and maximum iteration values. We see that the MALP success rate increases with increasing iterations, as expected. We see a similar trend in the increasing size of the largest connected component between MALP using rotate-at-s and normal rotate-at-s as well. Figure 12(b) shows the size of the largest connected component increasing with increasing iteration counts and ϵ value. In Figure 12(c), we compare the number of collision detection calls between MALP and rotate-at-s and we see similar computational cost increases as we saw using SL as the base local planner.

3. Path and Roadmap Clearance

To demonstrate MALP and the effects of deforming paths to the medial axis, we use a series of environments to build roadmaps and query them. We look at both query path and roadmap clearances by comparing roadmaps connected using SL and MALP and their query solutions.

a. Path Clearance

Path clearance experiments were performed using 2D (Figure 9(c)) and 8D (8 link version of Figure 9(f)) single homotopy class environments. Path clearance results are reported in Table I. Table I entries one and two are the 2D experiment and three and four the 8D experiment. In both cases, the query path extracted using MALP had both higher clearance and length than SL, showing us that the MALP paths are further away from obstacles and longer which is expected. Minimum and maximum path clearances are generally larger when using MALP as well.

Exploring multiple homotopy classes, we look at a 2D environment with 3 homotopy classes and the 3D clutter environment with multiple homotopy classes. In the 2D results, we see from columns 3 and 4 that MALP finds a path with much lower clearance and is shorter than SL path. Looking at Figure 9(d), we see the cause of this is that MALP finds a solution through the narrow passage where SL cannot. MALP can generate more connections than SL, allowing for different query solutions to be found. We see the same result in the many homotopy class 3D clutter environment Figure 9(b) MALP can generate maps which produce shorter medial axis paths than the medial axis path that can be found by a SL solution. This shorter path does not have any ensured increase or decrease average clearance, but for a path in that homotopic group, it has maximal clearance.

Table I. Statistics of different query solution paths. The set of entries (first through fourth) are for single homotopy class environments, the second set (fifth through eigth) are for multiple homotopy class environments. Paths are extracted from roadmaps using the same set of configurations connected using either MALP or SL local planner.

Homotopy Classes	Local Planner	Query Average Clearance	Query Length	Minimum Clearance	Maximum Clearance
1 (2D)	SL	0.3916	4673	0.0021	0.9499
	MALP	0.5246	5213	0.0039	1.0345
1 (8D)	SL	0.0344	101	0.0032	0.0864
	MALP	0.0466	120	0.0090	0.0857
3	SL	1.0255	3515	0.2202	1.3229
	MALP	0.2370	2745	0.0024	0.9513
Many	SL	3.6486	4667	0.0410	7.9096
	MALP	4.5572	2584	0.0187	11.121

b. Roadmap Clearance

Roadmap clearance experiments were performed on the the same environments as the path clearance experiments. The roadmap clearance results are reported in Table II. Roadmap clearance (third column of Table II) is calculated as the average clearance of all configurations, both nodes and edges, of a roadmap. The roadmap length (fourth column of Table II) is the sum of all the edge lengths in the roadmap. In our experiments the number of intermediate configurations along an edge is considered the length. Minimum and maximum clearance values are over all edges in the roadmap and the average minimum and maximum values are the averages of the minimums/maximums of all the edges.

Table II. Statistics of different roadmaps generated using MALP and SL local planner. The set of entries (first through fourth) are for single homotopy class environments, the second set (fifth through eighth) are for multiple homotopy class environments.

Homotopy Classes	Local Planner	Average Roadmap Clearance	Roadmap Length	Minimum Roadmap Clearance	Maximum Roadmap Clearance	Average Edge Min	Average Edge Max
1 (2D)	SL	0.6017	143052	0.0021	1.0324	0.4662	0.7096
	MALP	0.6498	177334	0.0021	1.0415	0.4670	0.6831
1 (8D)	SL	0.0474	7092	0.00016	0.0950	0.0304	0.0548
	MALP	0.0485	9246	0.00002	0.0950	0.0485	0.0559
3	SL	1.0002	158358	0.0002	1.3402	0.9954	1.1207
	MALP	0.9908	178122	0.0024	1.3517	0.9706	1.0800
Many	SL	4.5031	100240	0.0410	12.1546	3.0351	6.7148
	MALP	4.5385	254814	0.0048	12.0344	2.3678	7.0778

Looking at columns 3 and 4, we see that MALP generates roadmaps generally with higher average roadmap clearance and always longer in length compared to SL. When looking at average minimum, we see that the single homotopy class environments see and increase when looking at SL versus MALP, where as multiple homotopy class environments see a decrease. We attribute this to the multiple homotopy class environments and the ability MALP has of finding successful paths in tighter areas of C_{space}. MALP may find a path in these areas, but would reduce the overall clearance of the roadmap.

4. History Length

Here we explore the effect of the number of rays used for approximation and the history length on approximate medial axis retraction. We first generate medial axis samples in the 2D environment seen in Figure 9(e) using various ray count (5, 10, and 20) values and history lengths (5, 10, 20, and 40) using approximate medial axis retraction. We then measure the obstacle clearance of the sample and report the average clearance of the generated nodes in Figure 13. We would like to see the average clearance increase as we increase both history length and ray count.

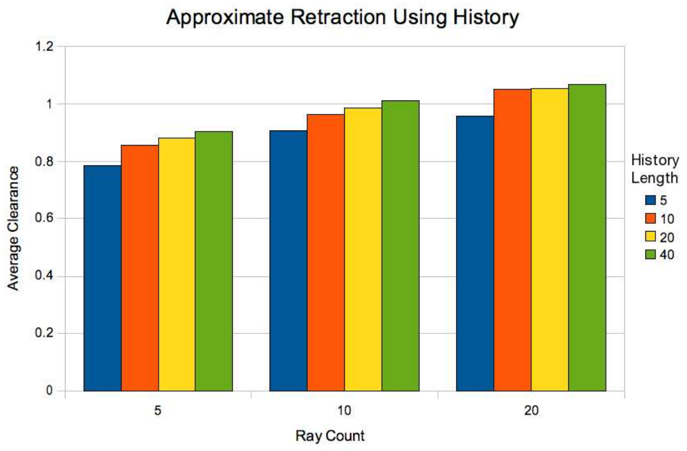


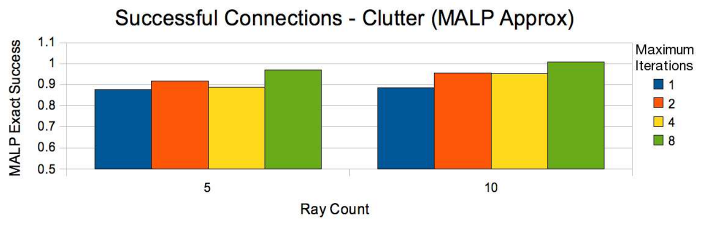
Fig. 13. Comparing the average distance away from the medial axis samples are using different history lengths (5, 10, 20, and 40) and ray counts (5, 10, and 20) in the 2D simple environment (Figure 9(e)).

From the Figure 13 we see that history length does impact retraction. As history length increases, the average obstacle clearance increases with the same ray count. Higher ray counts have better approximations since there is not as much noise to signal the premature finding of a peak. Thus, we conclude that history length reduces the effect of false positives caused by noisy approximate clearance computation and this benefit is greater with noisier approximations.

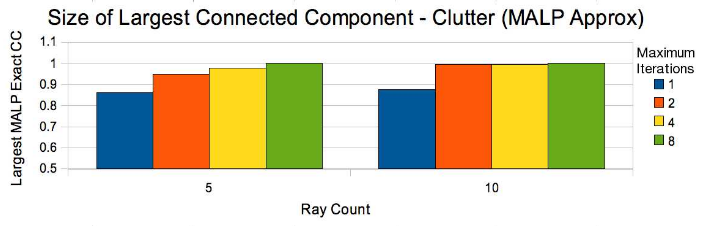
5. MALP Using Approximate Retraction

a. MALP for 3 Degrees of Freedom

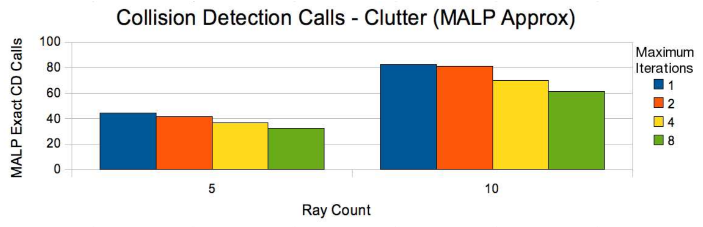
Figure 14 displays the same experiment seen in Figure 11, but using approximate C_{space} clearance computations instead of exact. Here, results for the MALP successful connection attempts, largest connected component size and collision detection calls are all normalized against MALP using exact clearance computation.



(a) Successful Connection Attempts



(b) Size of Largest Connected Component



(c) Number of Collision Detection Calls

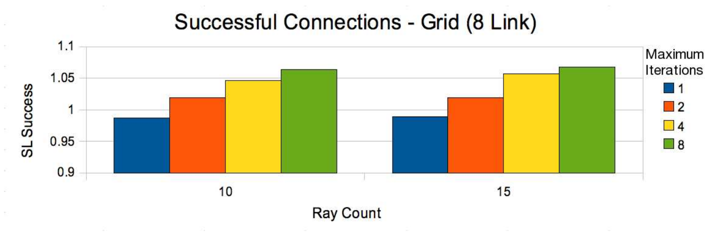
Fig. 14. Results from the 3D clutter environment (Figure 9(b)) using MALP with 1, 2, 4 and 8 iterations and approximate clearance computations using 5 and 10 rays with a history length of 20. The results are normalized against the performance of MALP using exact clearance computation.

In Figure 14(a) we plot the success of MALP relative to SL versus different ray counts and maximum iteration values. We see that approximate calculations perform almost as well as exact, with more 10 rays finding more connections than 5 rays, as expected. MALP using approximate clearance computation found generally %90 or more of the connections MALP using exact clearance computation found. We also see in Figure 14(b) that increasing the number of rays (i.e., improving the approximation) yields larger connected components and are relative in size the the ones found using MALP with exact clearance computation. Approximating C_{space} is not cheap and the cost is compounded by the retraction process, making approximate medial axis retraction quite expensive, see Figure 14(c).

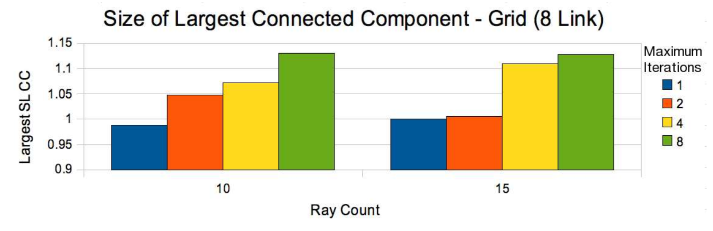
We note the collision detection calls relative to MALP exact decrease as iterations increase. We suspect this is attributed to the increased error using approximate clearance calculation in high clearance regions. False positives can be found earlier if initial configurations are already close to the medial axis. This seems to only be an artifact seen in the collision detection calls, connectivity does not seem to be effected. As seen in Figure 14(b), approximate retraction does not degrade with higher maximum iterations counts. Thought the cost may be expensive, in many cases, such as in non-convex environments or high degree of freedom problems, some or all of the clearance computations cannot be computed exactly and approximate methods must be used.

b. MALP for Higher Degrees of Freedom

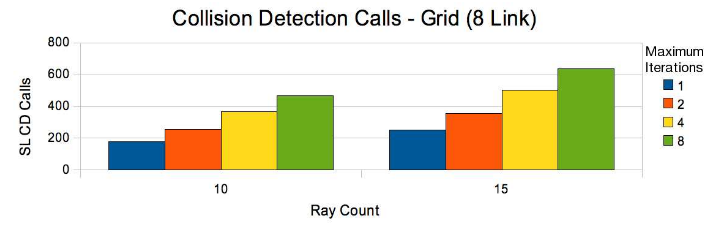
To analyze MALP in higher dimensions, we use an articulated linkage robot with only internal degrees of freedom. We connect nodes using MALP with approximate clearance computation and compare the performance to SL since MALP with exact clearance computation cannot be used. Figure 15 shows the results which are normalized on SL performance.



(a) Successful Connection Attempts



(b) Size of Largest Connected Component

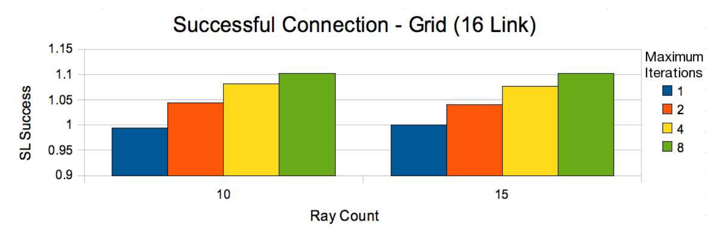


(c) Number of Connected Components

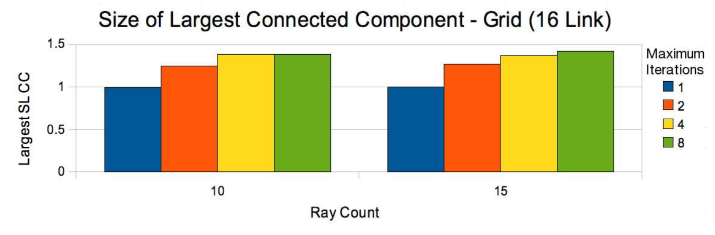
Fig. 15. Results from the 8D clutter environment (Figure 9(f)) using MALP with 1, 2, 4 and 8 iterations, and approximate clearance computations using 10 and 15 rays and a history length of 20.

We see in Figure 15(a) that MALP finds more connections as maximum iterations increases and performs better with with more rays, as expected. As with the other experiments, we see a increasing trend in the size of the largest connected component as the number of iterations increases (Figure 15(b)). Again, we see the cost of using MALP is expensive relative to SL as seen in Figure 15(c).

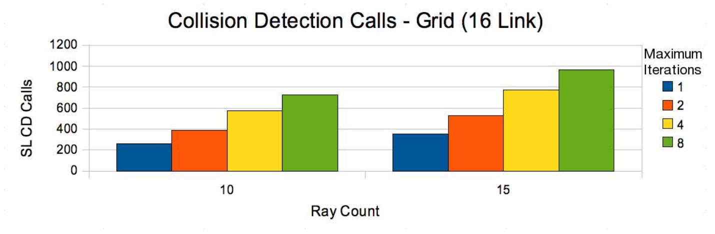
We see similar trends in the 16D results as we did in the 8D results lookin at Figure 16. Connection counts increase, as seen in Figure 16(a), and largest connected component size increases, as seen in Figure 16(b), as the maximum number of iterations



(a) Successful Connection Attempts



(b) Size of Largest Connected Component



(c) Number of Collision Detection Calls

Fig. 16. Results from the 16D clutter environment (Figure 9(f)) using MALP with 1, 2, 4 and 8 iterations and approximate clearance computations using 10 and 15 rays and a history length of 20.

increases. Looking at all the approximate experiments, we can state that higher ray counts perform better than lower counts due to the increase in accuracy. As we have mentioned, approximating C_{space} is not cheap, as can be seen looking at Figure 16(c), but still follows the general trends of MALP using exact clearance computation seen in Figure 11(c). However, approximate clearance computation is the only way to calculate clearance in higher degrees of freedom. MALP is expensive, but we see benefits in percentage of successful attempts. In practice MALP should be used in conjunction with a cheaper local planner if there are no restrictions on medial axis planning.

CHAPTER V

CONCLUSION

In this work we introduce MALP, a new local planner which deforms local planning paths to the medial axis. MALP works by taking a path between two medial axis configuations and determines if that path is ϵ-close to the medial axis. If the path is ϵ-close, a successful connection is found. If not, the middle point of that path is retracted to the medial axis and the 2 new paths, start to retracted middle and retracted middle to end, are recursed upon up to a maximum number of iterations. MALP, though costly, guarantees that edges are ϵ-close to the medial axis, thus greatly improving the quality of roadmap edges by increasing their clearance to obstacles. We have demonstrated MALP's use in 2D, 3D, 8D and 16D environments using both exact and approximate clearance computations.

Our results show that MALP out-performs SL local planning in terms of both connection success rate and size of largest connected component, and works in high dimensional problems. MALP produces roadmaps that not only have increased clearances over SL, but also have improved connectivity, a major challenge to PRM methods. The generality of MALP is reinfored by the definable base local planner. We have also shown that history length is a useful tool for improving the accuracy of approximate medial axis retraction.

MALP is more expensive than naive techniques such as SL but is more successful at connecting difficult to connect nodes and may in some cases find the key connection to bridge two connected components together. In practice MALP may be used in conjunction with cheaper local planners, such as SL, when maximal connectivity is priority.

REFERENCES

[1] J. H. Reif, "Complexity of the mover's problem and generalizations," in *Proc. IEEE Symp. Foundations of Computer Science (FOCS)*, San Juan, Puerto Rico, October 1979, pp. 421–427.

[2] H. Chang and T. Y. Li, "Assembly maintainability study with motion planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1995, pp. 1012–1019.

[3] Amit P. Singh, Jean-Claude Latombe, and Douglas L. Brutlag, "A motion planning approach to flexible ligand binding," in *Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, 1999, pp. 252–261.

[4] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, vol. 12, no. 4, pp. 566–580, August 1996.

[5] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo, "OBPRM: An obstacle-based PRM for 3D workspaces," in *Robotics: The Algorithmic Perspective*, Natick, MA, 1998, pp. 155–168, A.K. Peters, Proc. Third Workshop on Algorithmic Foundations of Robotics (WAFR), Houston, TX, 1998.

[6] V. Boor, M. H. Overmars, and A. F. van der Stappen, "The Gaussian sampling strategy for probabilistic roadmap planners," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 1999, vol. 2, pp. 1018–1023.

[7] D. Hsu, T. Jiang, J.H. Reif, and Z. Sun, "Bridge test for sampling narrow passages with probabilistic roadmap planners," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2003, pp. 4420–4426.

[8] Jory Denny and Nancy M. Amato, “Toggle PRM: Simultaneous mapping of C-free and C-obstacle - a study in 2D -,” in *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, San Francisco, California, USA, 2011, pp. 2632–2639.

[9] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, “MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1999, vol. 2, pp. 1024–1031.

[10] J.-M. Lien, S. L. Thomas, and N. M. Amato, “A general framework for sampling on the medial axis of the free space,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, September 2003, pp. 4439–4444.

[11] T. Lozano-Pérez and M. A. Wesley, “An algorithm for planning collision-free paths among polyhedral obstacles,” *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, October 1979.

[12] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, “Motion planning for a rigid body using random networks on the medial axis of the free space,” in *Proc. ACM Symp. on Computational Geometry (SoCG)*, 1999, pp. 173–180.

[13] C. Holleman and L. E. Kavraki, “A framework for using the workspace medial axis in prm planners,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2000, pp. 1408–1413.

[14] Mark Foskey, Maxim Garber, Ming C. Lin, and Dinesh Manocha, “Sm01-144: A voronoi-based framework for motion planning and maintainability applications,” Tech. Rep., University of North Carolina, 2001.

[15] H. Choset and J. Burdick, “Sensor-based exploration: The hierarchial generalized voronoi graph,” *Int. J. Robot. Res.*, vol. 19, no. 2, pp. 96–125, 2000.

[16] Howie Choset, Sean Walker, Kunnayut Eiamsa-Ard, and Joel Burdick, “Sensor-based exploration: Incremental construction of the hierarchical generalized voronoi graph,” *Int. J. Robot. Res.*, vol. 19, no. 2, pp. 126–148, 2000.

[17] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo, “Choosing good distance metrics and local planners for probabilistic roadmap methods,” *IEEE Trans. Robot. Automat.*, vol. 16, no. 4, pp. 442–447, August 2000.

[18] Y. K. Hwang and P. C. Chen, “A heuristic and complete planner for the classical mover’s problem,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1995, pp. 729–736.

[19] D. Hsu, J-C. Latombe, and R. Motwani, “Path planning in expansive configuration spaces,” *Int. J. Comput. Geom. & Appl.*, vol. 3, pp. 2719–2726, 1997.

[20] R. Geraerts and M. H. Overmars, “Reachablility analysis of sampling based planners,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2005, pp. 406–412.

[21] Sean Quinlan and Oussama Khatib, “Elastic bands: Connecting path planning and control,” in *In Proc. of the International Conference on Robotics and Automation*, 1993, pp. 802–807.

[22] Oliver Brock and Oussama Khatib, “Elastic strips: Real-time path modification for mobile manipulation,” *International Symposium of Robotics Research*, vol. 8, pp. 5–13, 1997.

[23] Roland Geraerts and Mark H. Overmars, “Creating high-quality paths for motion planning,” *Int. J. Robot. Res.*, vol. 26, no. 8, pp. 845–863, 2007.

[24] Leonard Jaillet and Thierry Siméon, "Path deformation roadmaps: Compact graphs with useful cycles for motion planning," *I. J. Robotic Res.*, vol. 27, no. 11-12, pp. 1175–1188, 2008.

[25] R. Geraerts and M. H. Overmars, "Creating high-quality roadmaps for motion planning in virtual environments," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2006, pp. 4355–4361.

[26] Eiichi Yoshida and Fumio Kanehiro, "Reactive robot motion using path replanning and deformation," in *ICRA*, 2011, pp. 5456–5462.

[27] S. Gottschalk, M. C. Lin, and D. Manocha, "OBB-tree: A hierarchical structure for rapid interference detection," *Comput. Graph.*, vol. 30, pp. 171–180, 1996, Proc. SIGGRAPH '96.

VITA

Name:	Kasra Mehron Manavi
Address:	c/o Dr. Nancy M. Amato Department of Computer Science and Engineering 3112 TAMU Texas A&M University College Station, TX, 77843
Email Address:	kmmanavi@gmail.com
Education:	B.S., Computer Science, University of New Mexico, May 2009 B.S., Applied Mathematics, University of New Mexico, May 2009
Membership:	AISES Member
Awards:	NSF Graduate Research Fellowship Honorable Mention

Publications:

1. Sam Ade Jacobs, Juan Burgos, Kasra Manavi, Jory Denny, Shawna Thomas, Nancy M. Amato, "A Scalable Method for Parallelizing Sampling-Based Motion Planning Algorithms," *To Appear* Int. Conf. on Robotics and Automation (ICRA), May 2012.

2. Samuel Rodriquez, Jory Denny, Juan Burgos, Aditya Mahadevan, Kasra Manavi, Luke Murray, Anton Kodochygov, Takis Zourntos, Nancy M. Amato, "Toward Realistic Pursuit-Evasion Using a Roadmap-Based Approach," Int. Conf. on Robotics and Automation (ICRA), May 2011.