Learning Motion Planning Assumptions

Anirudh Vemula Sanjiban Choudhury Sebastian Scherer

CMU-RI-TR-14-14

August 2014

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

The performance of a motion planning algorithm is intrinsically linked with applications that respect the assumptions being made. However, the mapping of these assumptions to actual environments is not always transparent. For example, a gradient descent algorithm is capable of tackling a complex optimization problem if some assurance of absence of bad local minimas can be ensured - however detecting the local minimas beforehand is very challenging. The state of the art technique relies on an expert to analyze the application, deduce assumptions that the planner can leverage and subsequently make key design decisions.

In this work, we make an attempt to learn a mapping from environments to specific planning assumptions. This paper presents a diverse ensemble of planners that exploit very different aspects of the planning problem. A classifier is then trained to approximate the mapping from environment to performance difference between a pair of planners. Preliminary results hints at the role played by convexity, whilst also demonstrating the difficulty of the classification task at hand.

Contents

1 Introduction

The general motion planning problem is to find a dynamically feasible collision free path from start to goal. While this problem statement might vary with changing applications, this statement alone has led to a diverse set of solution strategies. At a very high level, strategies can be categorized as discrete grid search, randomized sampling and gradient based methods. Each of the above strategies branch out further and further to create a planning strategy tree. Every node of this tree represents an unique planning configuration that leverages a set of assumptions that allow it to dominate in a specific application domain.

The genesis of these assumptions usually occurs when a particular planning problem configuration presents itself. An expert in the field analyzes the problem, searches for patterns in the objective function or constraints that can be exploited and then makes key design decisions to engineer a planning strategy. For example, if the environment has sparse clutter, an optimization approach is a natural choice because the local minima of the value function about the nominal path is most likely to be the global minima (even if guaranteeing such a claim maybe impossible).

A truly autonomous adaptive system must be able to make these decisions by observing the environment. In other words, we require the mapping from the environment to the validity of the assumptions. How to effectively take advantage of the mapping is another problem of interest.

In this paper, we conduct a preliminary investigation of the problem by looking at the niche sub-problem of learning the mapping of environment to performance difference between a pair of planners using different strategies. The idea is that any performance difference between the two is indicative of the validity of the different assumptions that they are making.

There has not been a significant amount of work in learning motion planning assumptions. Recent work by Abbeel et al. [7] has focused on applying machine learning to intialize an optimization procedure. Previous work by Dey et al. [4] attempts to learn a good ordering of initialization that an optimization routine tries sequentially. It is interesting to note that [7] raises similar questions in feature selections as we do in this paper. Another related work, [5], is about trajectory prediction where learning algorithms are used to predict a good path in a new environment from a database of demonstrated trajectories.

In this paper we present preliminary results of the classification problem. While results indicate that the classification problem for planning performance is indeed very hard as compared to other domains, they do hint at the importance of convexity as a feature. A key challenge that we encounter in this work is to find features that can suitably encode the properties of an environment that affect the planning problem.

2 Problem Statement

The problem statement is composed of two parts - the motion planning problem and the classification problem.

Let $X \subset \mathbb{R}^n$ be the configuration space of the system. Let $X_{obs} \subset X$ be invalid configurations that result in collision with obstacle. The dynamics of the system is specified as a dynamics constraint $g(x, \frac{dx}{dt}, \ldots, \frac{d^r x}{dt^r}) \leq 0, x \in X$, where r is the relative degree. Let the trajectory $\xi : [0,1] \rightarrow X$ be a smooth mapping from time to configuration. The planning problem is to find the shortest dynamically feasible trajectory from start x_0 to the goal x_f that is collision free. This is expressed as follows:

$$\begin{aligned} \underset{x(t)}{\text{minimize}} \quad & \int_0^1 \|\dot{x}(t)\| \mathrm{dt} \\ \text{subject to} \quad & x(0) = x_0 \\ & x(1) = x_f \\ & g(x, \frac{dx}{dt}, \ldots, \frac{d^r x}{dt^r}) \leq 0 \\ & x(t) \in X \setminus X_{obs}, t \in [0,1] \end{aligned} \tag{1}$$

Let $\mathcal{P}$ be an ensemble of planning strategies designed to solve the motion planning problem. Each algorithm is allocated a constant computation budget T_{comp}. For the scope of this work, the only results of interest is whether planner $\mathcal{P}_i$ is able to solve the problem within T_{comp}.

Let environments E be drawn from the distribution $\Gamma(E)$. A pair of planners $\mathcal{P}_i, \mathcal{P}_j$ is selected for comparison. The classification problem of interest is a 4-class problem defined by the distribution $\mathbb{P}_{i,j}$ over $\mathbb{X} \times \mathbb{Y}$, where $\mathbb{X}$ is some environment and $\mathbb{Y} = \{1, \ldots, 4\}$ is the label space. The labels correspond to the possible permutations of success and failures of $\mathcal{P}_i$ and $\mathcal{P}_j$. The goal is to find a classifier $c_{i,j} : \mathbb{X} \rightarrow \mathbb{Y}$ minimizing the classification loss on $\mathbb{P}_{i,j}$ given by

$$e(c_{i,j}, \mathbb{P}_{i,j}) = \mathbf{Pr}_{(x,y)\sim\mathbb{P}_{i,j}}[c_{i,j}(x) \neq y] \tag{2}$$

3 An Ensemble of Planning Approaches

The ensemble of motion planning strategies $\mathcal{P}$ is applied on a collection of environment $E_1, \ldots, E_N$ to solve Eq 1. At the end of the computation time budget T_{comp}, the binary success or failure of a planner $\mathcal{P}_i$ is assigned as a label.

In order for to have meaningful $\mathbb{P}_{i,j}$, the planner ensemble is designed to have diverse planning strategies that leverage fundamentally different assumptions. For the scope of this work, 3 such planners are selected - Heuristic Guided RRT*, Focussed RRT* and CHOMP. The following subsections goes into detail about the planning strategies and highlights the assumptions they make.

3.1 Heuristic Guided RRT*

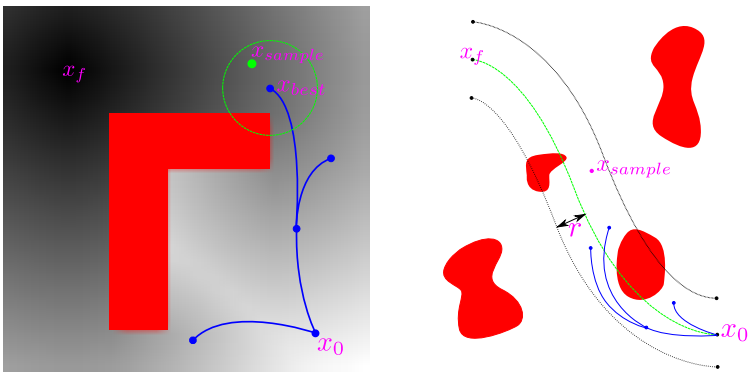
RRT* [6] is an asymptotically optimal motion planning algorithm that relies on random sampling in the configuration space and maintaining a search tree that is optimal in the limit. An attractive property of this approach is that it can arrive at globally optimal answers without the requirement of excessive parameter design. On the other hand, a significant disadvantage is the computation run-time of the approach. As a standalone approach, the RRT* makes minimal assumptions about the problem - as a consequence it fails to achieve a significant performance benefit over other approaches. It is popular to augment different functions in the RRT* with speedup techniques that are suitable for the application being considered [2].

The heuristic guided RRT* is a variant that augments the sampling function of RRT*. To compute an admissable "cost to go" and simplify heuristic computation, the constraint $g(x, \frac{dx}{dt}, \ldots, \frac{d^r x}{dt^r}) \leq 0$ is removed from Eq 1. This reduced holonomic motion planning problem can be solved using Dijkstra or Fast Marching Method on a grid of dimension n. During the growth of the RRT* tree, the sampling distribution is a Gaussian centred around the leaf node that has the least value - sum of the cost from parent and the heuristic cost. Algorithm 1 shows the outline of the approach.

The implicit assumption of this approach is that the solution of the reduced motion planning problem lies in the same proximity as the solution to the original problem. This assumption is violated in scenarios where obstacles create a bottleneck - in such scenarios the sampling misguides the tree growth into regions where a solution cannot exist.

Algorithm 1 Heuristic Guided RRT*

Parameters
  Δ : Grid resolution for dijkstra
  σ_b : Sampling variance around best leaf
Variables
  D : A grid storing cost to go
  x_{best} : The leaf node with least cost + heuristic
procedure HEURISTIC GUIDED RRT*(x_0, x_f)
  $V \leftarrow \{x_0\}, E \leftarrow \emptyset$
  $D \leftarrow$ DIJKSTRA$(x_0, x_f, X_{obs}, \Delta)$
  $x_{best} \leftarrow x_0$
  while TIMELEFT(T_{comp}) **do**
    $G \leftarrow (V, E)$
    $x_{sample} =$ SAMPLEGAUSSIAN(x_{best}, σ_b)
    $(V, E) \leftarrow$ EXTEND$(G, x_{sample}))$
    if COST(x_{sample}) $+D(x_{sample}) <$ COST(x_{best}) $+D(x_{best})$ **then**
      $x_{best} \leftarrow x_{sample}$
end procedure



(a) (b)

Figure 1: (a) Heuristic guided RRT* grows the search tree along the direction of heuristic function. The heuristic is shown in grayscale with brightness proportional to value. The leaf vertex x_{best} is the vertex with the smallest sum of cost to come from parent and heuristic cost to go. A random sample x_{sample} is drawn from a radius around this point (b) Focussed RRT* grows the search tree in a tunnel around the nominal path. The nominal path is the analytically optimal path in absence of obstacles. x_{sample} is drawn from a Gaussian of variance r.

3.2 Focussed RRT*

The focussed RRT* is a variant of RRT* where the sampling is biased around the solution that arises by ignoring X_{obs}. The optimal trajectory ξ_0 is solved for in the absence of X_{obs}. During the RRT* execution, the sampling is focussed in a tunnel around ξ. The tunnel is a Gaussian whose variance is the "radius" of the tunnel R. Algorithm 2 explains the steps in detail.

The approach assumes that an optimal solution exists within a proximity of a solution that ignores obstacles. This assumption is broken when dense clutter exists around the trajectory that the planner is biased towards.

3.3 CHOMP

CHOMP [9] is a very commonly used gradient-based trajectory optimizer because of it's efficient design of objective function and invariance to parametrization artefacts. The vanilla version of CHOMP optimizes a smoothness and obstacle objective while being constrained at two endpoints. To apply to Eq 1, the smoothness penalty term is scaled till the constraints $g(x, \frac{dx}{dt}, \ldots, \frac{d^r x}{dt^r}) \leq 0$ is met.

The advantage of the optimization approach is that if a good local minima exists around the initial guess, the optimization can find it much faster than sampling based approaches. The failure cases are when bad local minima

Algorithm 2 Focussed RRT*

Parameters
    R : The radius of the tunnel
Variables
    ξ_0 : The optimal trajectory ignoring X_{obs}
procedure FOCUSSED RRT*(x_0, x_f)
    $V \leftarrow \{x_0\}, E \leftarrow \emptyset$
    $\xi_0 \leftarrow$ OPTIMIZETRAJECTORY(x_0, x_f)
    while TIMELEFT(T_{comp}) **do**
        $G \leftarrow (V, E)$
        $x_{sample} =$ SAMPLETUNNEL(ξ_0, R)
        $(V, E) \leftarrow$ EXTEND$(G, x_{sample}))$
end procedure

appear.

4 Learning Framework

4.1 Features

The workspace $X_w \subset \mathbb{R}^2$ for the scope of this paper is considered to be 2D. The features used in this paper do not attempt to characterize the dynamics constraints - thus the target function being approximated implicitly captures this.

The feature vector is expected to encode the obstacle distribution and density of the environment. Any intuition about the assumptions that the planners exploit should be injected into this process to create an appropriate basis.

We draw features from 2 inspiration sources - the planning literature and image classification literature. From a planning perspective - convexity and obstacle density around the nominal trajectory play a key role. On the other hand, the environment can also be viewed as an image - features such as histogram of oriented gradients (HOG) and downsampled pixels appropriately characterize the high-pass and low-pass features.

4.1.1 Convexity Features

Convexity of the environment plays an important role in the performance of sampling based approaches to make connections across the configuration space.

The environment is discretized into a grid of fixed resolution. Each cell is assigned a value of 2,1,-1 corresponding to whether the goal is not directly visible from the cell, is visible or if the cell is in obstacle. The concatenation of these values is the convexity feature vector. Figure 2(a) shows an illustration.

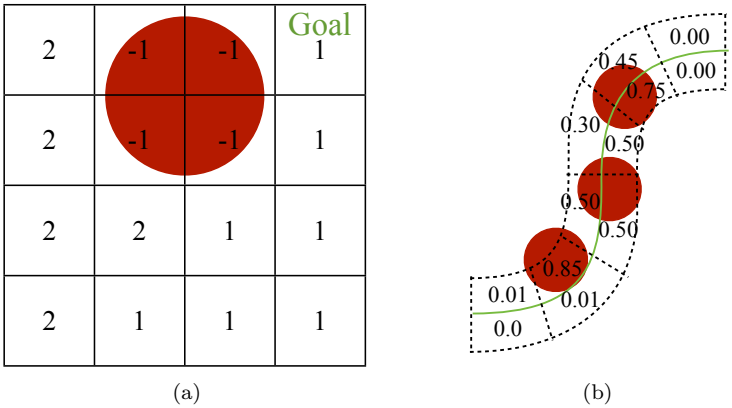


Figure 2: (a) Convexity feature vectory represents visibility to goal. The environment is discretized. Every cell is assigned a number 2,1,-1 corresponding to goal not visible, visible or cell in obstacle. (b) Tunnel density feature vector represents occupancy around the nominal trajectory. The volume around the nominal trajectory is discretized and obstacle density for each cell is computed.

4.1.2 Obstacle Density in a Tunnel around Nominal Trajectory

The obstacle density around the nominal path (the optimal path in absence of obstacles) encodes artifacts such as bad local minimas for gradient descent approaches and pitfalls for methods that focus search around the nominal.

A tunnel around the nominal path is discretized and the average density in every bin is used to create a feature vector. Figure 2(b) shows an illustration.

4.1.3 Histogram of Oriented Gradients

Histogram of oriented gradients [3] have historically had great success in image classification. It encodes gradient orientation in localized portions of an image (in our case, the binary image of the environment) thus characterizing the relative shape of obstacles.

4.1.4 Downsampled Pixels

To complement the high frequency character of HOG, a feature representing the overall density distribution of obstacles is required. This is achieved by downsampling the rasterized environment roughly to the resolution of other features.

4.2 Classification

In this work, we use Random Forests [1], an ensemble learning method popularly used for classification. It uses an ensemble of decision trees constructed at

training time and outputs the class that is the mode of the classes output by individual trees. Random Forests are apt for the classification in our approach as only a subset of the whole set of features is useful to distinguish a pair of planners.

5 Results

5.1 Experiment Setup

The planning problem is set for a 2 dimensional curvature constrained system. The configuration space X is a bounding box $[0, 50] \times [0, 50]$ with $x_0 = (2, 2)$ and $x_f = (48, 48)$. The constraint is set as $\frac{|\dot{x}\ddot{y} - \dot{x}\ddot{y}|}{(\dot{x}^2 + \dot{y}^2)^{\frac{3}{2}}} \leq \frac{1}{2}$. The environment is a poisson forest composed of random obstacles generated by using basic primitives like circles and polygons. The poisson density is varied uniformly from 0.005 to 0.01. In total 12000 environments were generated. 10000 are used for training and 2000 are set aside for testing.

5.2 Implementation Details

The motion planners have been implemented using Open motion planning library (OMPL), [10]. The RRT* implementation was augmented with the sampling procedure to create Heuristic Guided RRT* and Focussed RRT*. CHOMP is separately implemented adhering to the OMPL standards. The Heuristic Guided RRT* uses a resolution of 2 and the Focussed RRT* uses a tunnel variance of 5. CHOMP uses 500 waypoints with a regularization term of 0.01.

For feature extraction, a downsampling of scale 0.2 is used, the tunnel width considered is 2 and HOG features are extracted for every 10×10 block. For the Random Forest classifier, we have used Scikit-learn, a python machine learning package, [8].

5.3 Classification Performance

For brevity, the planner names are condensed to : HG-RRT* (Heuristic Guided), F-RRT* (Focussed RRT*). Table 1 shows the percentage accuracy obtained by various pairwise classifiers using different set of features on the collection of test environments. Note that the classifier predicts one of the four possible outcomes when two planners are run on the same environment. Observe that convexity features i.e. z_c play an important role in the classification of F-RRT* vs CHOMP and CHOMP vs HG-RRT* classifiers when compared to other standalone features. The degree of convexity of the environment depicts the effectiveness of a sampling based motion planner in such an environment. The two classifiers have a sampling based planner and an optimization based planner of which the sampling based is greatly affected by the convexity, hence convexity features display such an accuracy in these classifiers. Another interesting observation is

Classifier	z_d	z_t	z_c	z_h	z_{d+t}	z_{d+t+c}	$z_{d+t+c+h}$	z_{t+c}
HG-RRT* vs F-RRT*	48.3	42.1	45.7	41.7	52.4	**53.8**	51.1	49.3
F-RRT* vs CHOMP	46.6	45.2	49.0	44.5	49.7	**54.0**	52.6	50.1
HG-RRT* vs CHOMP	44.3	48.8	48.2	45.6	49.1	50.3	**53.2**	51.3

Table 1: Accuracy of the classifier in percentage using different set of features

z_d	Downsampling features
z_t	Tunnel features
z_c	Convexity features
z_h	HOG features

that tunnel features i.e. z_t have the highest accuracy in the CHOMP vs HG-RRT* classifier among other standalone features. This can be explained by the fact that CHOMP is an optimization based planner that optimizes around the initial trajectory and the initial trajectory chosen in our approach is the nominal path. Thus, the clutter in the tunnel around nominal path greatly affects the performance of CHOMP when compared to its effect on HG-RRT*, which is not significant. Hence, we see a high accuracy for tunnel features in this classifier.

5.4 Case Studies

In this section, we highlight a few examples that illustrate situations where assumptions of different planner failed. Examining these cases will provide vital clues about feature selection. In the following scenarios, if a sampling based planner is unable to find a path, the best leaf vertex and its path from root is shown. For CHOMP, the path with the least cost (even if its going through obstacles) is shown.

Figure 3 shows an example which highlights the pitfall of the heuristic. The heurisitic Dijkstra plans through a narrow opening which makes it very difficult for the RRT* to plan a feasible path through. The other planners fare better because they search around the nominal which is relatively collision free.

Figure 4 shows an example where heuristic guided RRT* is the only planner to solve the scenario. CHOMP's initial guess gets stuck in a bad local minima. Focussed RRT* has too tight a tunnel to maneuver around the obstacle.

Figure 5 shows an example where CHOMP is able to adjust its path to fit into narrow passages. Although it finds a suboptimal path in comparison to focussed RRT*, this example is indicative of the effect of the obstacle cost function and its gradient. Similar to Figure 3, the heuristic guided RRT* follows the heuristic

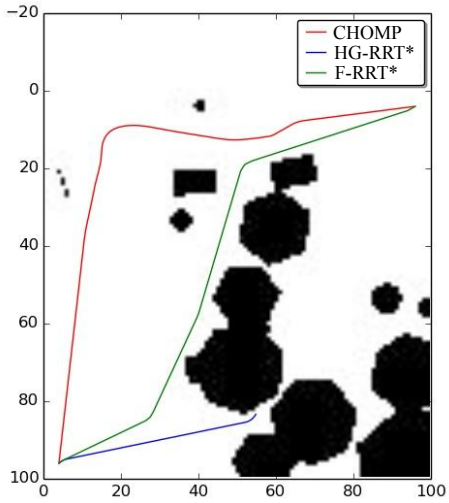


Figure 3: Heuristic Guided RRT* gets trapped as a result of ignoring dynamics constraints. Focussed RRT* is able to find a feasible path due to relatively clear tunnel. CHOMP finds a path as well, albeit more suboptimal.

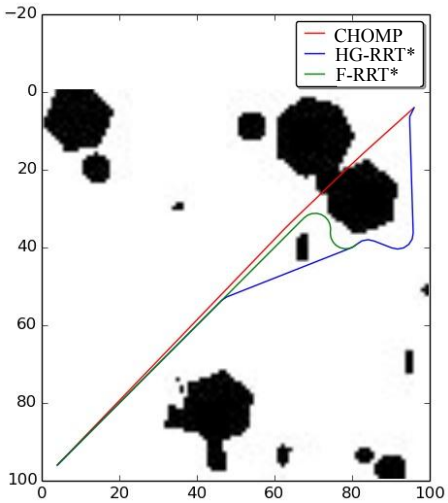


Figure 4: Only heuristic guided RRT* computes the best path. CHOMP gets stuck in a poor local minima. Focussed RRT* doesnt land samples that help it to maneuver around the obstacle.

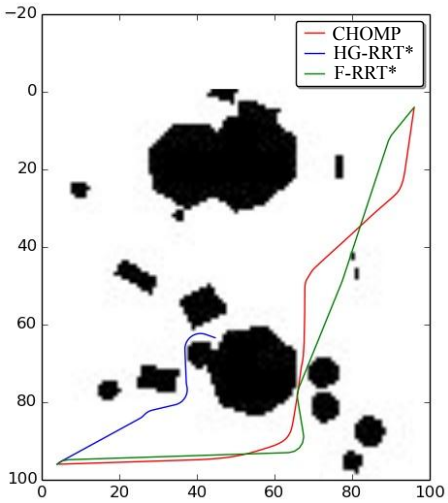


Figure 5: CHOMP is able to adjust to the shape of narrow passages. This is the effect of the obstacle cost and its gradients. Focussed RRT* grazes the obstacles due to its sampling based nature. Heuristic guided RRT* is led down a passage by the heuristic that the dynamics cannot pass through

into an area that the dynamics constraints prevent it from navigating through.

6 Conclusion

In this work, we have conducted a preliminary investigation on the mapping between assumptions made by a motion planning strategy and the environment configuration. Results indicate the role played by convexity and nominal path occlusion is significant in choosing the correct planning strategy. This make sense intuitively as well. Sampling based motion planning succeeds if it is able to make large connections across the configuration space which is measured by convexity. Optimization algorithms perform well when the shape of the value function around the optimization point has a good local minima which is indicated by the occlusion of the nominal path.

At the same time, the results fail to find out artifacts that seem obvious to the expert planner designer - presence of local minima and likely pitfalls of ignoring dynamics. These along with other factors lower the predicition accuracy of the classifier. Standard features that are well applicable to fields of image classification do not perform well in this paradigm. This is indicative of the larger problem of obtaining features that encode key aspects of the overall value function, which is intractable to compute given the large dimensionality of the problem.

References

[1] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[2] Sanjiban Choudhury, Sebastian Scherer, and Sanjiv Singh. Rrt*-ar: sampling-based alternate routes planning with applications to autonomous emergency landing of a helicopter. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3947–3952. IEEE, 2013.

[3] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[4] Debadeepta Dey, Tian Yu Liu, Martial Hebert, and J Andrew Bagnell. Contextual sequence prediction with application to control library optimization. *Robotics*, page 49, 2013.

[5] Nikolay Jetchev and Marc Toussaint. Fast motion planning from experience: trajectory prediction for speeding up movement generation. *Autonomous Robots*, 34(1-2):111–127, 2013.

[6] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[7] Jia Pan, Zhuo Chen, and Pieter Abbeel. Predicting initialization effectiveness for trajectory optimization.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[9] Nathan Ratliff, Matthew Zucker, J Andrew Bagnell, and Siddhartha Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 489–494. IEEE, 2009.

[10] Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012.