Proceedings of the 2005 IEEE
International Conference on Robotics and Automation
Barcelona, Spain, April 2005

Single-Query Entropy-Guided Path Planning

Brendan Burns Oliver Brock
Laboratory for Perceptual Robotics
Department of Computer Science
University of Massachusetts Amherst

***Abstract*— Efficient motion planning for robots with many degrees of freedom requires the exploration of a large configuration space. Sampling based motion planners perform approximate exploration of the configuration space in order to render the problem tractable. Each sample of configuration space as an opportunity to gain information about that configuration space. A formal definition of information gain can be used to guide a motion planner to achieve maximal progress toward the discovery of a path. We call such a motion planner *entropy-guided* since entropy reduction is synonymous with information gain. In the following we describe a single-query entropy-guided motion planner which uses a formal definition of information gain to focus its efforts on the acquisition of a single path from start to goal locations. Experimental evidence indicates that this approach can outperform existing single-query techniques.**

I. Introduction

Motion planning for robots with more than a few degrees of freedom is a challenging problem. In large part, this is because the size of the configuration space for such a robot is exponential in the degrees of freedom that the robot possesses. Even the general task of motion planning has been shown to be PSPACE complete [10] and complete exploration of a configuration space for a robot with more than a few degrees of freedom is computationally impractical. In the face of these exponential bounds, a planner must limit its exploration to relevant regions of the configuration space. Computation should not be expended on regions that provide little benefit to the motion planner.

Sampling-based motion planners reduce complexity by constructing an approximate model of the configuration space to plan with. Every sample taken results in the acquisition of information about the configuration space. Since each examination of a configuration provides information, and the examinations themselves are the primary computational expense incurred by a sampling based motion planner, it is preferable to select configurations which maximize the gain in information about the motion planner's task. *Entropy-Guided* motion-planning [4] uses a formal definition of information gain to guide exploration of configuration space for multi-query motion planning. In the following we describe a single-query entropy-guided motion planner.

Single-query motion planners limit exploration of the configuration space by focusing on a single path between a specified pair of configurations. Areas of configuration space that don't contribute to the discovery of this path are irrelevant to the planner. In our terms they offer no information to be gained. Existing single-query approaches approximate the relevance of a region of configuration space using a simple proximity heuristic. Regions that are closer to the start and goal points of the path query are judged to offer more information about the query than more distant configurations. Distant regions are only explored if a path through nearby regions proves to be impossible.

In contrast to proximity, the entropy-guided approach uses a formal definition of information gain to guide the motion planner's exploration. To do this, we first define a distribution that has high entropy when we have no information about the single-query and low entropy when we have found a path from start to goal. Sampling and exploration used by the motion planner is designed to maximize the expected reduction in the entropy, or information gain, of the distribution as a result of the knowledge obtained by the examination. By doing this, the planner is assured that given the current information available to it about configuration space, each action that it takes results in maximal progress toward discovering the path that has been queried. The entropy-guided planner chooses explorations that result in greater information gain and progress toward a solution, as a result we show an implementation of single-query entropy-guided motion planning outperforms existing single-query approaches.

II. Related Work

The first probabilistic approach to single-query path planning was the LazyPRM algorithm [2]. LazyPRM samples initially into the configuration space without performing any collision checks. Samples are assumed to be free and are connected to their nearest neighbors by edges without verifying their validity. LazyPRM searches using the A* algorithm which biases search toward the region of space surrounding straight line paths between the start and goal state. When a candidate path in the roadmap is found, it is validated by testing all nodes and then all edges. If obstructed nodes or edges are found, they are removed from the graph and A* path search begins again. A multi-grid variant of LazyPRM [1] discretizes the range of motion for each degree of freedom to simplify the configuration space. The granularity of the discretization is adapted until the motion planner can find a path.

Fuzzy PRM [9] developed simultaneously with LazyPRM and shares many of its characteristics. Unlike LazyPRM, Fuzzy PRM does not delay the examination of nodes in its graph, but does delay evaluation of edges. Fuzzy PRM maintains an estimated probability value for each edge based on a distribution over the unchecked portion of the edge. Candidate paths through configuration space were found using Dijkstra's algorithm with the obstructed probability of the edge as the edge weight. When a path is found, it is verified by examining edges in the order from longest (judge by FuzzyPRM to be least probable) to shortest (most probable).

Similar to the grid extensions of LazyPRM is the single-query quasi-random grid approach (LazyQRM) [7]. Quasi-random grids establish a lattice of configurations spanning the configuration space. Since the structure of the grid is implicitly defined, the costly pre-sampling and graph construction required by LazyPRM is not required. The A* algorithm using Euclidean distance to goal as its heuristic is used for search within the grid. The complexity of the A* algorithm is determined by the branching factor of each node in the search tree. For LazyQRM this value is given by the dimension of the configuration space and consequently the complexity of LazyQRM grows exponentially in the degrees of freedom.

An adaptive approach to single-query path planning is presented in [12]. It featured a meta-planner incrementally trying to plan between start and goal. At each planning attempt the "best" algorithm, based upon the number of obstructed configurations and the algorithm's previous performance, is selected and attempts a plan between the trees growing from the start and goal. When planning fails, any progress made by the planner is grafted onto the start and goal tree.

Rapidly-growing random trees (RRTs) [6] quickly explore the area between start and goal configurations by diffusing random trees of short edges through configuration space. Single-query planning with expansive spaces [5] also uses diffusion from start and goal configurations to find a path.

Others have suggested the use of information theory for motion planning. Yu and Gupta [13] use reduction in entropy to guide exploration. In their case, the entropy is of the model of the physical workspace and information is gained by sensing the real world, whereas we are interested in the entropy of the model of configuration space and gain information by sampling configuration space.

The notions of entropy and information gain have been used [3] to successfully guide sampling in the construction of multi-query probabilistic roadmaps. In this paper we use related techniques to minimize the number of configuration space examinations necessary for the construction of a feasible path between specified start and goal configurations.

III. Entropy-Guided Single-Query Motion Planning

To efficiently plan given the general computational complexity of motion planning, a motion planner must only explore a region of configuration space in proportion to the underlying importance of that region. To determine which areas of configuration space are most relevant or important, the motion planner needs information about the specific problem instance it is planning for. This information about the configuration space is used to determine which regions are most important to explore. Traditional single-query approaches to motion planning focus on configuration space regions near to start and goal configurations or along the shortest path connecting the configurations. This proximity heuristic gives the planner a definition for the relevance used to guide the motion planner.

In contrast, entropy-guided methods for motion planning [3] are inspired by the observation that each examination of configuration space obtains information about the motion planner's task. Each examination of configuration space should maximize the expected gain in information that results. In contrast to the entropy-guided approach to multi-query motion planning, single-query entropy-guided motion planning attempts to maximize information gain concerning the particular path we are searching for. An additional contribution of the single-query entropy-guided approach is that all explorations are also incorporated into an approximate model of configuration space. This model generalizes information from individual examinations and can be used to obtain predictions about the state of unobserved configurations.

In the following we discuss this approach in detail. First we give a formal definition of information gain for the single-query motion problem. Next we show how this can be used to guide exploration of configuration space in a concrete implementation. This implementation requires the development of a memory-based approximate model of configuration space.

A. Information gain for single-query planning

Information gain [11] is a formal representation of the reduction in uncertainty that results from some additional knowledge. It was originally proposed to formally model information transfer through electronic signals. In the case of sampling-based motion planning, additional knowledge is the observation that a configuration is obstructed or free. Previously, we defined information gain for multi-query motion planning [3]. For single-query motion planning we must define expected information gain for the task of discovering a particular path.

Entropy is the measure of uncertainty of a probability distribution P over a domain D:

$$H(D) = -\sum_{d \in D} P(d) \log P(d)$$

Information gain is the reduction in the entropy of a distribution as a result of obtaining some information i:

$$IG(D|i) = H(D) - H(D|i)$$

A distribution that has low entropy when a path between start and goal has been found allows information gain to be used to direct exploration. At each step of the motion planner, taking steps that maximize information gain (and thus minimize entropy) of this distribution results in maximal progress toward a solution to the specified path query.

For single-query motion planning we use a distribution over a set of possible paths A in a roadmap R. Each member of this set $a \in A$ represents a path connecting the start and goal configurations. The probability assigned to each path a in this distribution is the probability that it will be the successful path returned by the motion planner. This probability is the combination of the probability that the path is free ($P_f(a)$) and the probability that this path will be examined by the motion planner prior to any other path which is free ($P_s(a)$). Since these probabilities are independent, the joint probability that the path is free and examined prior to any other free path is given by:

$$P_p(a) = P_s(a) \times P_f(a)$$

The probability, $P_s(a)$ is difficult to calculate exactly but it is proportional to the length of the path since the planner uses A* which searches for shortest paths.

$P_f(a)$ can be calculated as the product of the probability that it's constituent vertices $V(a)$ and edges $E(a)$ are free:

$$P_f(a) = \left(\prod_{v \in V(a)} P_f(v)\right)\left(\prod_{e \in E(a)} P_f(e)\right)$$

Edge and vertex probabilities are either the result of direct observation in the collision checker or estimated by the approximate model (Section IV-A).

The entropy of this distribution is given by:

$$H(D) = -\sum_{a \in A} P_p(a) \log P_p(a)$$

Every exploration of configuration space results in obtaining of some new information i which pertains to the feasibility of the path a.

For each path a in A, there are two possible outcomes of learning i: a may be more likely to be free, or a may now be known to be obstructed. In each case, the information gain is given by the difference between the prior and current entropy. Most of the probabilities for the paths of the distribution will remain the same, only those paths that contain a vertex or edge related to i will be affected. Let A' be this set of all paths in A that contain paths affected by i .

First, consider the case where i results from an observation that something is free. In this case, the probability of each path that i pertains to increases slightly:

$$\begin{aligned} IG(D|i) &= H(D) - H(D|i) \\ &= -\sum_{a \in A'} P_p(a) \log P_p(a) - -\sum_{a \in A'} P_p(a|i) \log P_p(a|i) \end{aligned}$$

When i results from an obstructed observation path $a \in A'$ that i pertains to, the probability of the path becomes zero. The information gained is:

$$\begin{aligned} IG(D|i) &= H(D) - H(D|i) \\ &= -\sum_{a \in A'} P_p(a) \log P_p(a) - -\sum_{a \in A'} P_p(a|i) \log P_p(a|i) \\ &= -\sum_{a \in A'} P_p(a) \log P_p(a) \end{aligned}$$

Expected information gain is given by:

$$\begin{aligned} < IG(D|i) > \;=\; & -\sum_{a \in A'} P_p(a) \log P_p(a) + \\ & P(i = \text{free}) \sum_{a \in A'} P_p(a|i) \log P_p(a|i) \end{aligned}$$

Observing that $\log P_p(a|i) \leq 0$ and that $P_p(a|i) \geq 0$ for any path a, we can see that for information pertaining to a set of paths A', the information gain from discovering an obstruction is greater than or equal to information gain for observing free space. Intuitively this can be seen by noting that observing a configuration is obstructed immediately eliminates the entire path, while observing a configuration is free only increases the probability the path is free.

The following section describes how this information theoretic analysis of single-query motion planning can be used to develop an single-query entropy-guided motion planner.

IV. A Single-Query Entropy-Guided Planner

The formal definition of information gain previously presented leads to the development a single-query entropy-guided motion planner. The exact computation of information gain is not feasible to compute for a number of reasons, including the difficulty of assessing $P_s(a)$ the probability of a candidate path a being selected prior to any free path and the computational expense of calculating the set A' of paths whose probability is influenced by knowing i. Despite this, the formal definition of information gain for single-query motion planning can be used to guide the operation of a single-query motion planner. At each stage of the motion planner's operation, intuitions from the information gain calculation can be used to shape the planner's behavior.

At the initial stage of motion planning, the single-query entropy-guided motion planner chooses an initial set of samples from which an initial roadmap and approximate model of configuration space are constructed. A fraction of the samples are chosen uniformly at random, while the majority are chosen from the hyper-cube bounding box surrounding the start and goal configurations. To build the model, all sampled configurations are inspected by the collision checker to determine if they are free or obstructed. Details of the model are given later (Section IV-A). The initial roadmap is constructed from configurations which are found to be free, but without verifying the connecting edges. This is one important difference between the entropy-guided approach and LazyPRM [2] which constructs an initial roadmap without examining any configurations. Another important distinction is that while LazyPRM samples quite heavily and uses short edges to connect configurations, we sample sparsely and connect using longer edges.

Once the initial roadmap has been constructed, A* is used to find a candidate path between start and goal configurations. In contrast to previous uses of A* for path planning [1], [2] that use edge length for edge cost, the cost used by entropy-guided planning is the product of edge length and the probability the edge is obstructed:

$$\text{Cost}(e) = k \times \text{Length}(e) \times P_f(e)$$

The probability that an edge is obstructed is estimated by the approximate model. This cost is designed to favor edges that are likely to be free, while simultaneously maintaining a focus on the goal node. The constant k is used to balance this trade-off. A*'s heuristic g remains the Euclidean distance to the goal node, this also ensures that search is directed toward the goal configuration. This cost heuristically attempts to maximize $P_s(a)$ and $P_f(a)$ which in turn maximizes the information gain resulting from either validating or eliminating the path a. $P_s(a)$ is maximized because the path is short it is likely to be chosen by A*. $P_f(a)$ is maximized because paths which are likely to be free are favored.

Once a candidate path is found, the algorithm begins by examining each of its vertices. The expected information gain from examining a vertex is greater than for examining an edge since the set of paths A' affected by gaining information about a vertex is greater than the set of paths affected by gaining information about an edge. Because observing an obstructed vertex provides more information than observing a free vertex, the vertices are examined in order according to their probability of obstruction. If any vertex is obstructed, examination of the candidate path stops and search for a new candidate path resumes. This scheme for examining paths is unlike the LazyPRM approach which proceeds from start and goal nodes toward the middle of the path.

Once all vertices in the candidate path are verified, the edges of the candidate path are examined. Again the edges are examined in order by their probability of obstruction. If an edge is found to be obstructed it is removed and search for a new candidate path resumes. If all edges are found to be free, the path is the solution.

If a candidate path between start and goal cannot be found in the graph, it is necessary to enhance the roadmap to introduce new candidate paths. The resampling that we use is similar to that of LazyPRM, we resample configurations that are near to obstructed edges that connect valid configurations in the roadmap. Unlike LazyPRM, we filter the configurations that we resample through the approximate model of configuration space. If a resampled configuration is likely to be obstructed, we do not add it into the roadmap. The planner learns from experience and avoids pathologically resampling the same invalid regions of configuration space. Once resampling is performed, the search for a candidate path continues using the augmented roadmap.

Throughout all of the operations of the single-query entropy-guided planner it is necessary to be able to quickly obtain estimates of a configuration or edge's state without the computational expense of examining it in a collision checker. To accomplish this we build an approximate memory-based model of the pertinent configuration space using the information acquired by the planner so far.

A. Models

Although previous single-query work can be seen to have built implicit models of configuration space, the use of expected information gain requires a more sophisticated model. In particular it depends on predictions of the state of unobserved configurations. The approximate model also incorporates information that existing single-query methods discard. In particular it retains information from obstructed samples in order to predict that nearby configurations are likely obstructed as well.

Modeling configuration space can be viewed as a classical machine learning classification problem [8]. An approximate model is constructed from a training set of configurations which have been examined in a collision checker and observed to be free or obstructed. The approximate model uses these observed configurations to approximate the state of unobserved query configurations.

For our approximate model we use a nearest neighbor model [8]. The nearest neighbor model is a local, memory-based approximate model. It estimates the state of an unobserved configuration by examining the set of k nearest neighbors in the model. The state (observed or free) with the greatest number of nearby neighbors is take as the state of the unobserved configuration. We have shown elsewhere [4], that approximate models can build successful approximations of configuration space.

Nearest neighbor models have a number of appealing characteristics for our purposes. First, adding data to the model takes constant time regardless of the size of the model. Also, querying the model is linear in the number of

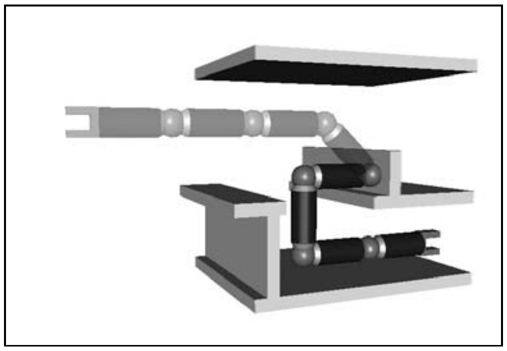


Fig. 1. The initial (transparent) and final (solid) configuration of a twelve degree of freedom arm in the experimental environment.

configurations used to construct it. Additionally, the model does not attempt to model the entire configuration space. Since we only add configurations that are relevant to the single-query, only a model of the regions related to the particular path is built. Finally, since nearest neighbors is a local model, it easily adapts to intricate local configuration space topologies such as narrow passages and peaks.

V. Experiments

To validate the entropy-guided approach to single-query motion planning we perform experiments with the implementation of a single-query entropy-guided planner described previously (Section IV). The performance of the entropy-guided planner is compared to the performance of traditional LazyPRM [1] and the single-query quasi-random planner (LazyQRM) [7]. Initial parameters for these two algorithms are set based upon descriptions in the respective papers.

To compare the algorithms we measure the number individual calls to the collision checker, the number of calls to validate an edge and the total overall time to find a path. For the entropy-guided planner, the number of collision checks used to construct the initial roadmap and model is included in the total number of collision checks.

Experiments were run for an arm with six, nine and twelve degrees of freedom. The twelve degree of freedom arm is shown in Figure 1. The six degree of freedom arm consisted of three links connected by joints with two degrees of freedom. The nine degree of freedom arm used the same three links but joints with three degrees of freedom. The twelve degree of freedom arm added a fourth link and joint with three degrees of freedom to the nine degree of freedom arm. The workspace for all of the arms is the same and is shown in Figure 1. Each algorithm runs ten times with ten different path queries. Each path query consisted of a random starting location in the vicinity of the straight configuration shown in Figure 1 and a goal configuration with the end effector inside the constrained location in workspace (also pictured in Figure 1). In this way, none of the paths queried were trivially easy for the motion planner to solve. All of the path queries represented the most challenging trajectory in the workspace.

The results of the experiments are shown in Figure 2. It can be seen that entropy-guided single-query planning outperforms the other two planners. The LazyQRM planner fails to complete for either the nine or twelve DOF worlds. It consumes all available memory and exits on a Pentium 4, 3.2Ghz with 1 gigabyte of RAM. This is indicative of the fact that LazyQRM's grid grows exponentially in the dimensionality of the configuration space.

It is important to note that in the twelve degree of freedom world neither the LazyPRM nor the entropy-guided approach could reliably find a path. The LazyPRM planner successfully found a path 50% of the time and the entropy-guided planner found a path 75% of the time. In these successful attempts, LazyPRM outperforms entropy-guided, but it is successful less often. These results are summarized in Table I. When LazyPRM is successful, it is because it has selected beneficial placements for its initial roadmap. Thus, its performance is biased by optimal placements. Entropy-Guided motion planning is less reliant on receiving a good initial roadmap and can find solutions more often. However, some of the solutions take more effort to discover. We intend to explore extensions to the entropy-guided approach, including the use of better approximate models, and resampling techniques to address this issue.

For the purposes of practical motion planning we apply a time cut-off to each algorithm. Any path planning attempt that lasted longer than thirty seconds is halted and path planning restarts from the beginning. The collision and edge checks as well as the runtime are all accumulated until a successful motion plan occurs. The graphs in Figure 2 show these results. The graphs indicate that the entropy-guided approach leads to better runtimes for all three problems. The greater number of collision checks in six and nine degrees of freedom are from the checks used to build the initial roadmap and model. This constant cost is subsumed by motion planning collision checks for the twelve degree of freedom arm. It is important to note that individual collision checks require an order of magnitude less computation than edge checks, so the slight differences in the number of collision checks has much less of an effect on runtime than the number of edge checks. Additionally single-query entropy-guided motion planning is biased toward checking edges which are likely to be obstructed, while LazyPRM is biased toward edges likely to be free. Obstructed edges are generally less computationally difficult to check than free edges, resulting in further performance gains.

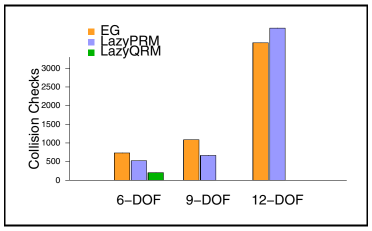
VI. Conclusions

Motion planning for robots with many degrees of freedom is a problem with proven complexity. To practically plan despite these general bounds on performance an

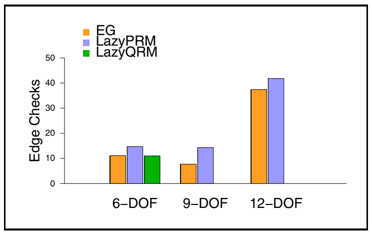
Algorithm	Success	Collision Checks	Edge Checks	Runtime
Entropy Guided	75%	2433.1	26.1	5.9
LazyPRM	50%	1575.5	19.25	5.0
LazyQRM	0%	N/A	N/A	N/A

TABLE I

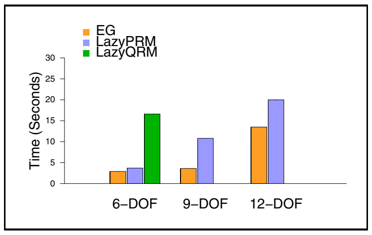
PERCENTAGE OF SUCCESSFUL MOTION PLANS FOR THE 12-DOF ROBOT



(a) Average collision checks



(b) Average edge checks



(c) Average runtimes

Fig. 2. Experimental results for motion planning in 6, 9 and 12 degree configuration spaces

efficient motion planner must maximally exploit available information about a particular problem instance to adapt computation toward the solution to this particular problem.

Each examination of configuration space obtains information pertinent to the discovery of a path. Entropy-Guided motion planning is a sampling-based approach to motion planning in which every configuration sampled by the planner is chosen to provide the maximal expected gain in information related to the task of the motion planner. In the preceding we have described an entropy-guided approach to sampling-based single-query motion planning which maximizes the information gain concerning a particular path through configuration space. We have given a formal definition of information gain for the single-query problem and shown how this definition can be used to develop a single-query entropy-guided motion planner.

We have also demonstrated the use of an approximate model of configuration space which allows the motion planner to acquire more information about configuration space from each observed sample of configuration space. This model is more sophisticated than a traditional roadmap. It can provide estimates of the state of configurations that have not been observed, based upon input from nearby configurations. Each examination of configuration space refines the prediction of expected information gain and suggests new configurations to examine next. An important distinction of this planner in comparison to existing approaches is that it introspects its progress *while* motion planning and uses this information to guide future exploration.

Experimental results with an implementation of this single-query entropy-guided motion planner show that it is capable of outperforming existing approaches to single-query motion planning.

REFERENCES

[1] R. Bohlin. Path planning in practive: Lazy evaluation on a multi-resolution grid. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 49–54, Maui, USA, 2001.

[2] R. Bohlin and L. E. Kavraki. Path planning using lazy PRM. In *Proceedings of the International Conference on Robotics and Automation*, volume 1, pages 521–528, San Francisco, USA, 2000.

[3] B. Burns and O. Brock. Information theoretic construction of probabilistic roadmaps. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 650–655, Las Vegas, 2003.

[4] B. Burns and O. Brock. Model-based motion planning. Technical Report TR 04-32, Computer Science Department, University of Massachusetts Amherst, 2004.

[5] D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *International Journal of Computational Geometry and Applications*, 9(4):495–512, 1999.

[6] S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report TR 98-11, Iowa State University, 1998.

[7] S. M. LaValle and M. S. Branicky. On the relationship between classical grid search and probabilistic roadmaps. In *Proceedings of the Workshop on the Algorithmic Foundations of Robotics*, Nice, France, 2002.

[8] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[9] C. L. Nielsen and L. E. Kavraki. A two level fuzzy PRM for manipulation planning. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 1716–1722, Takamatsu, Japan, 2000.

[10] J. H. Reif. Complexity of the mover's problem and generalizations. In *Proceedings of the Symposium on Foundations of Computer Science*, pages 421–427, 1979.

[11] C. E. Shannon. A mathemtatical theory of communication. *Bell System Technical Journal*, 27:379–423, July 1948.

[12] D. Vallejo, I. Remmler, and N. M. Amato. An adaptive framework for single shot motion planning: A self-tuning system for rigid and articulated robots. In *Proceedings of the International Conference on Robotics and Automation*, pages 21–26, Seoul, Korea, 2001.

[13] Y. Yu and K. Gupta. An information theoretic approach to viewpoint planning for motion planning of eye-in-hand systems. In *Proceedings of the International Symposium on Industrial Robotics*, 2000.