Measuring Guidance for the Motion Planning Decision Problem

Amnon Attali, Stav Ashur, Nancy M. Amato

Department of Computer Science, University of Illinois
201 N. Goodwin Avenue, Urbana, IL 61801, USA
aattali2@illinois.edu, stava2@illinois.edu, namato@illinois.edu

Abstract

Cost-to-go, i.e., the length of the shortest path, is the standard metric for computing distance between configurations in robotics. In this work we propose an alternative metric for measuring distance between configurations for the *decision version of the problem*, in which the goal is to find any path between configurations, if one exists. We show intuitively and experimentally that if one aims to find any path, and not just the shortest, then our visibility based metric is a more suitable measure than cost-to-go. Moreover, we show that unlike other visibility based measures, ours can be approximated easily in higher dimensions using the same techniques as in randomized sampling algorithms.

Introduction

In the motion planning problem we are given a robot r, an environment (workspace) E, and two configurations s and t of r in E, and are tasked with finding a valid sequence of movements, i.e. actions that can be executed by the robot and do not cause any collisions, that take r from configuration s to t.

By defining the *configuration space* (Lozano-Pérez and Wesley 1979) as the implicit space of all configurations of r in E, and partitioning it into *free space*, the set of configurations where r is fully within the boundary of E and does not collide with any obstacle (or with itself in the case of articulated robots), and *obstacle space* which is the complement of free space, we can reduce the motion planning problem to the (s, t)*-connectivity problem* (STCON) in c-space.

Much research effort has been dedicated to solving the optimization version of this problem, where the required output is the shortest path (according to some metric) between s and t or some approximation of it, see e.g. (Amato et al. 2000; Kuffner 2004). The algorithmic distinction between these problems can be seen by considering the well known RRT algorithm (Lavalle 1998) designed to solve the STCON problem, and its shortest path adaptation variant RRT* (Karaman and Frazzoli 2010). In this paper we only consider the STCON problem in c-spaces.

The metric we use to define the optimal distance (i.e., problem difficulty) between states has large implications for planning. For example, learning based methods often attempt to learn estimates of such metrics (Bhardwaj, Choudhury, and Scherer 2017; Chiang et al. 2020). Note that such methods almost universally attempt to learn cost-to-go based quantities even if they are not interested necessarily in finding the shortest path. Recent work (Attali et al. 2024) has formalized the notion of guidance in sampling based motion planning (SBMP) algorithms, which moves the discussion beyond metrics in c-space to how they are used by planning algorithms during search. This formal definition isolates the guidance component of SBMP algorithms by measuring the quality of the guidance with respect to some underlying target metric. However, most of these results are only for the shortest path variant of the motion planning problem, as can be seen by their definition of guidance quality based on cost-to-go.

In this paper we extend the results of (Attali et al. 2024) to the STCON problem by defining a geometric-visibility based distance metric as a building block for the evaluation of guidance quality.

We start by defining the visibility based distance metric, and then move on to show that it better correlates to the hardness of STCON motion planning problems than other reasonable metrics such as cost-to-go ($C2G$) and Euclidean distance. We use the runtime of RRT queries as a proxy to the hardness of an STCON instance.

Related work

Sampling Based Motion planning SBMP algorithms create geometric graphs in c-space by randomly sampling configurations as graph nodes, and connecting pairs of configurations by edges representing movements of the robot from the source node configuration to the target node's configuration. Most SBMP algorithms use only valid nodes, which are fully inside the environment's boundary and collision-free, and edges, where the validity of an edge is determined by the validity of intermediate configurations along the continuous motion represented by it.

In the RRT algorithm (Lavalle 1998) the c-space graph is a tree rooted at the start configuration node, and nodes and edges are added by a process of randomly sampling a configuration v, finding its nearest neighbor on the tree u, and extending an edge from u in direction $\vec{uv}$ until a predetermined maximum distance is travelled or a collision is detected. In contrast to RRT, Probabilistic Roadmap (PRM) algorithms are used for a multi-query motion planning scenario. PRMs randomly sample configurations to serve as graph nodes, and connect close-by pairs of nodes with edges. Given an (s,t) query, an attempt is made to connect s and t to some of their nearest neighbors, and then find an (s,t)-path in the roadmap graph.

Dense roadmaps can be used to approximate shortest paths between many (s,t) pairs, but the basic PRM algorithm can also be used to solve a single STCON instance. Given an (s,t) query first add s and t to the roadmap, then iteratively add uniformly sampled nodes to the graph, connect them to some of their nearest nodes, and return once s and t are in the same connected component.

Visibility Based Motion planning A PRM variant called Visibility-based PRM (Simeon, Laumond, and Nissoux 2000) (VBPRM) is an attempt at a "pure" STCON multi-query algorithm. Sampled nodes are only retained by the algorithm if they are not only valid, but also either not "visible" to any other node previously added, or visible to nodes that do not see each other. This geometric notion of of visibility, which we also use in this paper, defines the symmetric "seeing" relationship, i.e. two configurations c and c' in some space see each other if the line segment $\overline{cc'}$ does not intersect any interior point of an obstacle. In c-space the obstacle set is the set of connected components of the obstacle space.

It is also important to mention that the notion of visibility has been used in shortest path motion planning algorithms as well, mostly due to the fact that the shortest path for a point robot in a 2D environment is a path in the visibility graph of the environment's nodes. Many motion planning algorithms have therefore used visibility graphs as a tool (Vegter 1990; Wein, van den Berg, and Halperin 2005; You, Cai, and Wu 2019).

Guidance in Motion Planning The intuitive notion of guidance can be found in almost every motion planning algorithm in the form of human knowledge that has been "baked" into the different algorithms. Some obvious examples are DR-RRT (Denny et al. 2020) which samples close to a workspace skeleton, e.g. the workspace medial axis, and Lazy PRM (Bohlin and Kavraki 2000) which attempts unvalidated paths based on their length, which is initially equivalent to using the Euclidean distance as guidance.

Some algorithms have a less clear underlying guidance. In RRT, for example, we extend the node closest to a randomly sampled point. Upon closer inspection we can see this means that the node chosen to be expanded is chosen with probability equal to the measure of its Voronoi cell in c-space, a bias known as Voronoi Bias.

In recent work (Attali et al. 2024) there has been an attempt at formalizing this notion of motion planning guidance as the sampling bias of the algorithm, and measuring an algorithm's guidance quality by defining a target sampling distribution and computing the distance of the empiric distribution generated by the algorithm, i.e. the distribution that emerges from the sampling performed by the algorithm, to that target distribution.

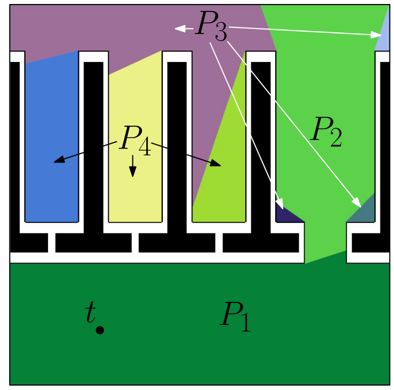

Figure 1: An illustration of the window partitioning of a 2D c-space of a rectangular robot (axis aligned rectangle with no rotational DOF) with respect to a configuration t. The levels of the hierarchy are denoted with P_i, where $P_0 = \{t\}$, and the boundaries between levels are the windows. Notice that the 3rd and 4th levels of the hierarchy are composed of disjoint polygons. The black upside down T-shapes are the workspace obstacles, and the white areas are obstacle space computed by performing a Minkowski sum of the robot r with the obstacles.

Preliminaries

The geometric notion of visibility is very intuitive when considering polyhedrons in Euclidean space. Given a polyhedron $P \subseteq \mathbb{R}^d$, we say that two points $p, q \in P$ *see* each other if the line segment $\overline{pq}$ lies completely within P, or formally, $\overline{pq} \cap P = \overline{pq}$, and the *visibility polyhedron* of p, denoted $V(p)$, is the set $\{q \in P \ : \ p \text{ sees } q\}$. Translating this to the motion planning landscape, we consider the free space as our polyhedron. Two configurations see each other if the local planner is able to connect them with a valid motion, and the visibility polyhedron of a configuration is the set of all points that are reachable by such local plans. We similarly define the visibility polyhedron of a set $S \subseteq P$ of points to be $V(S) = \bigcup_{p \in S} V(p)$.

The *visibility graph* of a polyhedron P usually refers to the graph over the nodes of P, where two nodes are connected by an edge if they see each other. This definition can be extended to include all of the points of P, meaning we get an infinite graph. We call the shortest path between two points p, q in the unweighted visibility graph the *link distance* between them, and denote it by $LD(p,q)$.

The *Window partitioning* (Suri 1986) of a polyhedron P with respect to a point $t \in P$ is a hierarchical partitioning of P that captures the link distances from every point of P to t. We define P_1 to be $V(t)$, and P_{i+1} to be $\{p \in P \mid p \in V(P_i) \setminus \bigcup_{j=1}^{i-1} P_j\}$. Simply put, this is the partition of P into sub-polyhedrons based on the link distance from t. The surfaces in the interior of P separating levels of the hierarchy are called *windows*. See Figure 1 for an example.

Method

In (Attali et al. 2024) sampling efficiency is defined as the negative log likelihood of an empirical sample with respect to some target distribution. Specifically, the target distribution (over nodes v of some search tree T) used for shortest path problems is

$$Q_T(v) = \frac{\exp(-\tau_v/\tau)}{\sum_{u \in T} \exp(-\tau_u/\tau)}$$

where τ_v is the cost-to-go from v to the goal state. In other words, a softmin of cost-to-go values over nodes of the tree with temperature parameter τ.

Our goal is to define a target distribution which we can use to measure sampling efficiency (quality of guidance) for the STCON problem, by proposing a new metric to replace cost-to-go in the above computation.

In the shortest path problem, $C2G$ provides a very convenient metric for deciding which node to expand as it is continuous and most pairs of nodes have different $C2G$ values such that movement in any direction affects the distance to the goal configuration in some distinct way.

This, however, is not the case in the STCON problem. If two configurations s and t are in the same connected component of free space, then any other point in that component and any continuous movement of s will never be "better" or "worse" than s itself, as all of these configurations reside within the same connected component. In other words, we are attempting to give different meaningful values to data points selected from a binary distribution.

In the rest of this section we describe the distance metric we have developed, and provide the reasoning behind it as well as algorithmic details. The metric is a function of two components, the link distance and visible window measure.

Link Distance

Consider the 2D environment illustrated in Figure 2. Topologically s, t_1, and t_2 are all inside the same region, and even though the distances of the shortest curves connecting them have a very similar length, our intuition says that s is somehow "more connected" to t_1 than to t_2. It is this intuition that stands at the core of the visibility based metric, and has motivated us to use the link distances captured by window partitioning.

Another natural candidate for a visibility based metric is the *shortest path link distance* (SP-LD), which is the number of edges in the shortest (s,t)-path. However, this metric, as exemplified by the (s, t_3)-path shown in Figure 2, does not match intuition in some relatively simple scenes. We can replace the octagon in the figure with an arbitrary polygon, and get (s, t_3)-paths of very similar lengths with an increasing number of line segments, while the intuitive connectivity of s and t_3 remains exactly the same.

Computing the window partitioning described in the previous section is a non-trivial operation in 2D, and prohibitively complicated in arbitrary dimensions. As such we propose using a simple discrete approximation. We use a c-space graph and iteratively compute the graph nodes that belong to the next level of the window partition hierarchy, as opposed to creating this partition for the entire continuous space. Algorithm 1 describes our basic method for computing this hierarchy, though note we have employed some simple heuristics and parallelized collision detection between segments and obstacles in order to improve the runtime, e.g., by batching edge detection calls for "*p sees q*" queries.

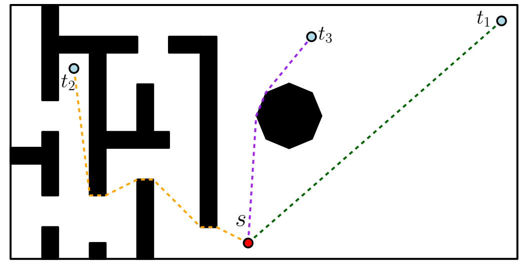

Figure 2: Illustration of the relevance of link distance to the STCON problem guidance. t_1 is intuitively more connected to s than t_2 (due to visibility, or link distance). At the same time, the exact shape of the convex obstacle between s and t_3 should intuitively not affect their visibility, showing that shortest path link distance is not a good metric.

Algorithm 1: Window Partitioning

Input: A goal configuration $t \in C$ (c-space)

$P \leftarrow$ A set of c-space points
$LD \leftarrow 1$
$prev\_level \leftarrow \{t\}$
$hierarchy \leftarrow \{(prev\_level)\}$
while $P \neq \emptyset$ **do**
  $new\_level \leftarrow \{\}$
  if $prev\_level = \emptyset$ **then**
    $new\_level \leftarrow P$
    $new\_level.ld \leftarrow \infty$
    $P = \emptyset$
  end
  else
    $new\_level.ld \leftarrow LD$
    for $q \in prev\_level$ **do**
      if p *sees* q **then**
        $new\_level \leftarrow new\_level \cup \{p\}$
        $P \leftarrow P \setminus \{p\}$
      end
    end
  end
  $hierarchy$.`insert`(new_level)
  $LD \leftarrow LD + 1$
  $prev\_level \leftarrow new\_level$
end
return $hierarchy$

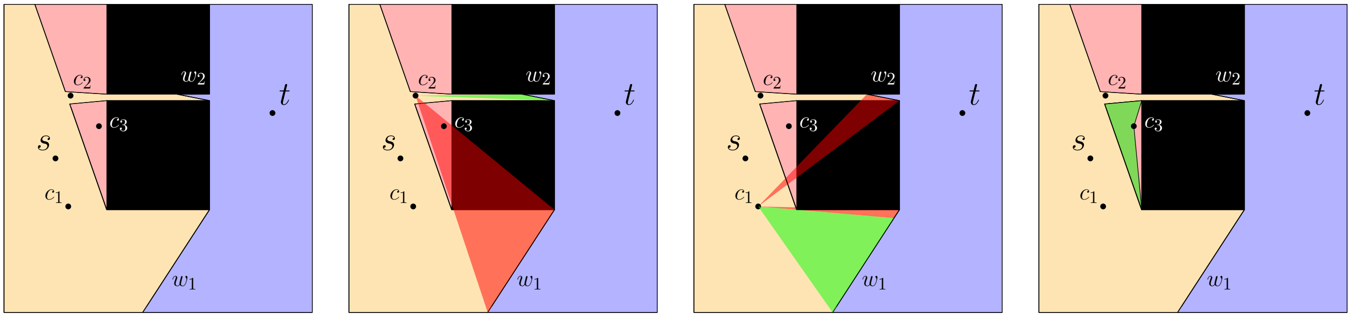

Figure 3: The left three images illustrate the relevance of the window "angle" component - even though c_2 is closer to the lower level of the hierarchy than c_1, it is a worse candidate for expansion since exploration through narrow passages is difficult. The rightmost image demonstrates the importance of monotonicity along a path, the relative value of having a large window measure (as does c_3) should be less than the value of moving down a level in the hierarchy. The directions leading to windows to the lower level of the hierarchy are colored green, obstructed directions are colored red.

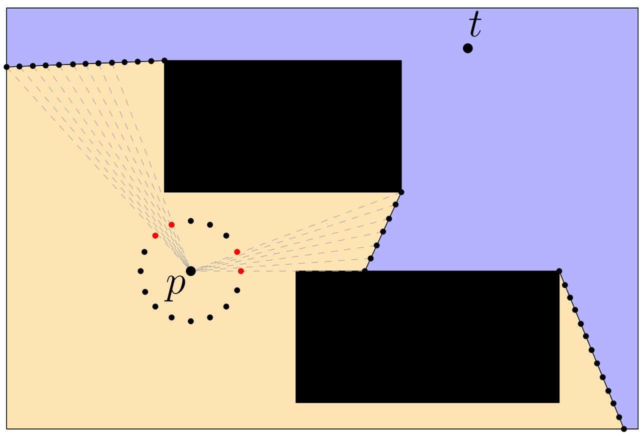

Figure 4: An illustration of the approximation algorithm for computing the window visibility measure in a 2D space for a given point p. The two level window partition contains three windows composed of several window nodes each. Line segments are drawn from each such node which is visible from p. A set of points on the 2D sphere centered at p are also shown, and four out of the 16 nodes are marked as a nearest neighbor of one of the segments giving the result $1/4$.

Algorithm 2: Visible Window Measure

Input: A configuration $c \in C$ (c-space),
A set S of points on the unit sphere,
A set of windows $W = \{w_i\}_{i=1}^{k}$

$S \leftarrow S + c$
$dirs \leftarrow \emptyset$
for $window \in W$ **do**
  for $p \in window$ **do**
    if `is_valid`($\overline{cp}$) **then**
      $dirs \leftarrow dirs \cup S$.`NearestNeighbor`($\overline{cp}$)
    end
  end
end
$S \leftarrow S - c$
return $\frac{|dirs|}{|S|}$

Visible Window Measure

How should we differentiate between the value of expanding two nodes in the same level of the hierarchy? Figure 3 provides intuition for why distance to the closest window of the lower level of the hierarchy does not imply a point is a better candidate for expansion. This intuition is addressed by the visible window measure component of our metric which takes into account the measure of directions that "lead" to a window in a lower level of the hierarchy. In 2D this corresponds to the measure of angles from which the first visible point outside the current level of the hierarchy is on the lower level. In other words, if a point is at level i, what is the percentage of angles such that a ray shot from that point at that angle leave level i directly to level $i-1$?

Computing the measure of directions leading to a window in a high dimensional space is non-trivial, but can be approximated using the graph nodes which separate consecutive levels of the hierarchy. We call these separating nodes window nodes, and can easily compute the window nodes of a level by running a multi-source search from the nodes of that level which terminates upon finding nodes in a different level. Algorithm 2 describes how we use such nodes to compute the visible window measure. Namely, we maintain a set of uniformly sampled points on the unit sphere S. When required to compute the measure of directions for some point p we can translate S to be centered at p, and then for every window node v, such that p sees v, find the nearest neighbor of the line $\overline{pu}$ on the sphere, and at the end of the process return the ratio of marked to overall sphere points. Figure 4 demonstrates this approximation visually.

Given that the link distance is a discrete function, we set the effect of the visible window measure component on the metric as a normalized value between 0 and 1. This ensures that the distance to the goal from configurations along a topological path to the goal would decrease monotonically. For example, configuration c_3 in Figure 3 has most of its "field of vision" occupied by a window leading to the lower level in the hierarchy. If the metric were to attribute a significant reduction in distance based on this large field of vision, then nearby states in the lower level of the hierarchy would be assigned a larger distance to the goal (due to their comparatively small visibility window measure). Consequently, a search algorithm using such a metric to explore would get stuck in the local minima around c_3.

Finally, since in most practical cases having a constant fraction of the directions lead to a desired expansion is sufficient for exploration, we do not reward visible window measures of more than $1/4$ the maximum field of view (e.g. $\pi/2$ in 2D). The visible window measure for a d-dimensional c-space with a goal configuration t is therefore defined as

$$wvm_t(c) = \frac{\min(\mu_{d-1}(WVD(c), \mu_{d-1}(\mathbb{S}^d)/4)}{\mu_d(\mathbb{S}^d)/4} \tag{1}$$

Where $WVD(c)$ is the set of directions, μ_{d-1} is some $(d-1)$-dimensional measure, and $\mathbb{S}^d$ is the d-dimensional unit sphere. Notice that $wvm_t(c)$ is a value in the interval $[0, 1]$.

STCON Metric Given a goal configuration t, our metric using the link distance and the visible window measure is defined as

$$LD_t^+(c) = LD(c,t) - wvm_t(c) \tag{2}$$

It is evident that our metric is unhelpful for STCON "No" instances, namely instances where s and t are in different connected components. This is another place where we see the challenges encountered by the binary nature of the problem, but as the main purpose of this metric is to measure the guidance of motion planning algorithms that almost always assume they are given a "Yes" instance, one can argue that in "No" instances every sample is equally useful, as in not at all useful, since it does not make any progress towards a solution.

Experiments

We show that the value given to (s,t) configuration pairs by visibility based metrics better correlates to the "hardness" of the STCON problem than the value generated by the Euclidean distance between s and t or the $C2G$ metric, i.e. the length of the shortest (s,t)-path. We do so by experimentally computing the correlation between five different metrics and an estimate for the difficulty of an STCON problem. We test three visibility based metrics, LD, LD^+, and shortest path link distance ($SP - LD$), and two widely used distance metrics $C2G$, and Euclidean distance. We use the runtime of RRT as a proxy for estimating the difficulty of an STCON problem.

We use a 2D environment with two mostly open spaces connected by two passages, one simple and relatively wide, and another jagged and relatively narrow (through which is the shortest path in the environment between the two rooms for most configurations). See Figure 5. While we purposefully chose this problem to highlight the suitability of our metric for the problem, we maintain that the existence of several homotopy classes of c-space curves with very different geometric properties, e.g. clearance and number of turns, between two points, is not a rare or pathological example.

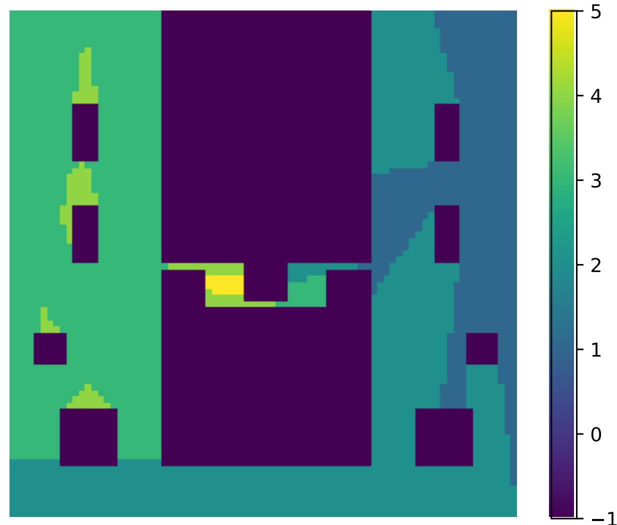

Figure 5: The environment (and also the c-space) for the experiment and its window partitioning. Obstacle space points are shown with a value of -1. Other colors indicate the (estimated) level in the window partition hierarchy of each state relative to a goal in the upper right region. This visualization was created by creating the point set in Algorithm1 as points on grid

We computed the (linear regression) correlation between the five metrics' values for a set of 500 (s,t) pairs, and the runtime of RRT for those tasks. We use RRT runtime as a proxy to the hardness of STCON problems since, to the best of our knowledge, other attempts to quantify this hardness have not been done.

Our results are shown in Table 1 and Figure 6. Notice that the r^2 values (explained variance) of the visibility based metrics SP-LD, LD, and LD^+ is significantly better than that of Euclidean distance, and while the values of $C2G$ and LD are very similar, $LD+$ produces better results than both and about the same as SP-LD which does not generalize to higher dimensions and, as shown in Figure 2, can be rather easily manipulated. We have removed a very small number of outliers with respect to RRT runtime (whose inclusion only strengthened the results provided here).

Metric	r^2 Value
$C2G$	0.178
SP-LD	0.253
LD^+	0.252
LD	0.175
Euclidean	0.088

Table 1: r^2 Values of the linear regression models of cost-to-go ($C2G$), shortest path link distance (SP-LD), the direction measure enhanced link distance (LD^+), link distance (LD), and Euclidean distance.

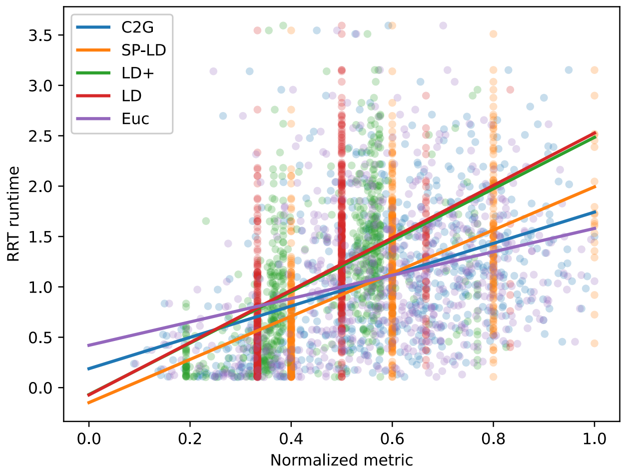

Figure 6: A plot of ≈2500 data points matching 500 RRT queries measured by 5 different metrics gathered in our experiment, and their linear regression models. The x-axis corresponds to the normalized metric value, and the y-axis corresponds to the runtime of an RRT query in seconds.

Discussion and Future Work

In this paper we have defined a new visibility based metric, which can be easily approximated in higher dimensions, for the decision variant of the motion planning problem, and provided evidence in support of the claim that it measures a relatively large portion of the hardness of the STCON motion planning problem.

We hope in future work to explore the effects of defining a sampling efficiency metric (one that is tree-relative rather than purely environment-relative) using LD^+ (instead of $C2G$). For example we could test the intuition that more effective STCON algorithms, while failing to explore the shortest path, do efficiently select nodes from which the goal is more visible. Finally, we also wish to test the hypothesis that our metric (and sampling efficiency based on our metric) could be more effective than cost-to-go for learning algorithms for motion planning problems.

References

Amato, N. M.; Bayazit, O. B.; Dale, L. K.; Jones, C. V.; and Vallejo, D. 2000. Choosing Good Distance Metrics and Local Planners for Probabilistic Roadmap Methods. *IEEE Trans. Robot. Automat.*, 16(4): 442–447.

Attali, A.; Ashur, S.; Love, I. B.; McBeth, C.; Motes, J.; Morales, M.; and Amato, N. M. 2024. A Framework for Guided Motion Planning.

Bhardwaj, M.; Choudhury, S.; and Scherer, S. 2017. Learning heuristic search via imitation. In *Conference on Robot Learning*, 271–280. PMLR.

Bohlin, R.; and Kavraki, L. E. 2000. Path Planning Using Lazy PRM. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 521–528.

Chiang, H.-T. L.; Faust, A.; Sugaya, S.; and Tapia, L. 2020. Fast Swept Volume Estimation with Deep Learning. In *Alg. Found. Robot. XIII*. Springer. (WAFR '18).

Denny, J.; Sandström, R.; Bregger, A.; and Amato, N. M. 2020. Dynamic Region-biased exploring Random Trees. In *Alg. Found. Robot. XII*. Springer. (WAFR '16).

Karaman, S.; and Frazzoli, E. 2010. Incremental Sampling-based Algorithms for Optimal Motion Planning. In *Proceedings of Robotics: Science and Systems*. Zaragoza, Spain.

Kuffner, J. J. 2004. Effective Sampling and Distance Metrics for 3D Rigid Body Path Planning. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation, ICRA 2004, April 26 - May 1, 2004, New Orleans, LA, USA*, 3993–3998. IEEE.

Lavalle, S. M. 1998. Rapidly-Exploring Random Trees: A New Tool for Path Planning. Technical report, Iowa State University.

Lozano-Pérez, T.; and Wesley, M. A. 1979. An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles. *Communications of the ACM*, 22(10): 560–570.

Simeon, T.; Laumond, J.-P.; and Nissoux, C. 2000. Visibility-Based Probabilistic Roadmaps for Motion Planning. *Advanced Robotics*, 14(6): 477–493.

Suri, S. 1986. A linear time algorithm for minimum link paths inside a simple polygon. *Computer Vision, Graphics, and Image Processing*, 35(1): 99–110.

Vegter, G. 1990. The Visibility Diagram: a Data Structure for Visibility Problems and Motion Planning. In Gilbert, J. R.; and Karlsson, R. G., eds., *SWAT 90, 2nd Scandinavian Workshop on Algorithm Theory, Bergen, Norway, July 11-14, 1990, Proceedings*, volume 447 of *Lecture Notes in Computer Science*, 97–110. Springer.

Wein, R.; van den Berg, J. P.; and Halperin, D. 2005. The Visibility-Voronoi Complex and Its Applications. In Mitchell, J. S. B.; and Rote, G., eds., *Proceedings of the 21st ACM Symposium on Computational Geometry, Pisa, Italy, June 6-8, 2005*, 63–72. ACM.

You, Y.; Cai, C.; and Wu, Y. 2019. 3D Visibility Graph based Motion Planning and Control. In *ICRAI 2019: 5th International Conference on Robotics and Artificial Intelligence, Singapore, November 22-24, 2019*, 48–53. ACM.