10-2020

Asymptotically-Optimal Topological Nearest-Neighbor Filtering

Read Sandström

Jory Denny
University of Richmond, jdenny@richmond.edu

Nancy M. Amato

Recommended Citation

Sandstrom, Read, Jory Denny, and Nancy M. Amato. "Asymptotically-Optimal Topological Nearest-Neighbor Filtering." *IEEE Robotics and Automation Letters* 5, no. 4 (October 2020): 6916–23. https://doi.org/10.1109/LRA.2020.3017472.

Asymptotically-Optimal Topological Nearest-Neighbor Filtering

Read Sandström[1] Jory Denny[2] Nancy M. Amato[3]

***Abstract*—Nearest-neighbor finding is a major bottleneck for sampling-based motion planning algorithms. The cost of finding nearest neighbors grows with the size of the roadmap, leading to a significant computational bottleneck for problems which require many configurations to find a solution. In this work, we develop a method of mapping configurations of a jointed robot to neighborhoods in the workspace that supports fast search for configurations in nearby neighborhoods. This expedites nearest-neighbor search by locating a small set of the most likely candidates for connecting to the query with a local plan. We show that this filtering technique can preserve asymptotically-optimal guarantees with modest requirements on the distance metric. We demonstrate the method's efficacy in planning problems for rigid bodies and both fixed and mobile-base manipulators.**

***Index Terms*—Motion and Path Planning, Semantic Scene Understanding**

I. Introduction

SAMPLING-BASED motion planners such as PRM [1] and RRT [2] explore a problem by sampling random configurations for a robot and connecting them to nearby neighbors. The nearness of two configurations is determined by a distance metric defined over the entire configuration space. An important component in the process of connecting configurations is a nearest-neighbor search, which determines which existing configurations to attempt connecting to a new sample. Nearest-neighbor finding is a major bottleneck for sampling-based planners. A brute-force scan requires linearithmic time for each connection attempt, while more advanced methods such as the k-d tree [3] approach logarithmic time with a sufficiently large number of configurations.

However, prior work has focused primarily on utilizing faster computational techniques as opposed to leveraging the shape of the free space. Such approaches rely on metrics which ignore visibility when computing point-to-point distance. In sampling-based planning, a nearest-neighbor must be visible to the query point in order to be connectable via a local plan. Metrics which are blind to the obstacle space consider only proximity, which can lead to failed connections in the presence of thin walls (Fig. 1(a)). When this occurs, the compute resources spent on nearest-neighbor finding are wasted.

Topological Nearest-Neighbor Filtering [4] presents a solution to this problem by considering the connectivity of the free configuration space, $\mathcal{C}_{free}$, when determining viable nearest neighbors. This method maps configurations to local neighborhoods in workspace that are described by a cell decomposition mesh. Given a query point q, configurations in q's neighborhood can be quickly located, and the mesh's adjacency relationships provide a means to quickly locate additional configurations in neighborhoods that are nearby through the connected workspace. This works in many robotics applications where the workspace has a large influence on the shape of $\mathcal{C}_{free}$, e.g., where a rigid body must navigate a tunnel. These candidates can be passed as an input set to another nearest-neighbor method.

In this work, we show how the filter can preserve the asymptotic-optimality guarantees of RRT* [5] and SST [6] by leveraging a relationship between the distance metric and workspace translation of each link. This is important because asymptotically-optimal planning can require large roadmaps to converge to desired path costs. The topological filter is an effective way to identify promising candidates for connection without considering the entire (large) roadmap. We also show how filtering can be applied to multi-link robots such as manipulators. We evaluate the filtering technique in asymptotically-optimal planning settings for a rigid-body robot, fixed-base manipulator, and mobile-base manipulator. This work represents a portion of the author's PhD dissertation [7].

II. Related Work

In this section, we describe relevant prior work on nearest-neighbor methods in sampling-based planning, asymptotically-optimal planners, and inner distance.

A. Nearest-Neighbor Methods

Nearest-neighbor finding has attracted significant attention in sampling-based motion planning research as a primary performance bottleneck. Two primary avenues have been investigated: computing the exact set of nearest neighbors, and bounded approximations of this set.

A popular method for exact nearest-neighbor finding in a motion planning context employs a k-d tree to quickly

Manuscript received: Feb., 24, 2020; Revised June, 2, 2020; Accepted June, 26, 2020.

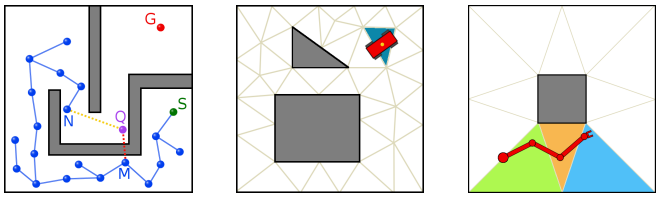
This paper was recommended for publication by Editor Dezhen Song upon evaluation of the Associate Editor and Reviewers' comments. This research supported in part by NSF awards CSF-1439145, CCF-1423111, EFRI-1240483, RI-1217991.

[1]Read Sandström is with the Parasol Laboratory, Dept. of Computer Science and Engineering, Texas A&M University, College Station TX 77840, USA. readamus@tamu.edu

[2]Jory Denny is with SpiRoL: Spider Robotics Lab, Dept. of Mathematics and Computer Science, University of Richmond, Richmond, VA, 23173, USA. jdenny@richmond.edu

[3]Nancy M. Amato is with the Parasol Laboratory, Dept. of Computer Science, University of Illinois Urbana-Champaign, Champaign, IL, 61820, USA. namato@illinois.edu

Digital Object Identifier (DOI): see top of this page.



(a) (b) (c)

Fig. 1. (a) An example scenario where proximity alone is not a good metric for choosing a nearest-neighbor. The planner is searching for a nearest-neighbor for the sample Q from the roadmap (blue). The best neighbor is N, but a pure proximity metric will choose M because it is closer in the configuration space $\mathcal{C}_{space}$. (b) A configuration of a rigid-body robot (red) is contained by the decomposition cell (blue) which contains its reference point (yellow). (c) A configuration of a multi-link robot (red) where the neighborhood cells are colored for the first (green), second (orange), and third (blue) joints.

compute the nearest neighbor [8]. A later work re-iterates on this method to develop a partitioning strategy which better respects the nuances of the orientation components [9]. The use of k-d trees is also observed for nearest-neighbor problems in machine learning. Other structures such as metric and cover trees have also been suggested, although empirical evidence suggests that these are not significantly faster than k-d trees in problems of moderate size and dimension [10].

However, other research has noted that k-d trees perform little better than brute-force when the problem dimension is moderate to large [11]. Such difficulties have spurred investigation in approximate nearest-neighbor methods, which aim to relax accuracy for a large gain in speed. Locality-sensitive hashing is a well-known class of approximate methods which attempts to bucket similar samples together with some form of hashing scheme, and has been applied in both machine learning [12], [13] and motion planning [11]. An alternative approximate method from the motion planning realm is distance-based projection onto Euclidean space (DPES), which measures distance to landmarks in a lower-dimensional projected space [14].

Topological Filtering [4] investigates a nearest-neighbor filter based on workspace connectivity. It attempts to select a small set of candidate neighbors which are near to the query point through some connected subset of $\mathcal{C}_{free}$, (e.g. workspace). The intention is to avoid choosing points that are nearby according to a distance metric, but are actually very far apart through $\mathcal{C}_{free}$ (Fig. 1(a)).

B. Asymptotically-Optimal Planners

Another area of interest in sampling-based motion planning is the study of *asymptotically-optimal* (AO) planning algorithms. These techniques continue to refine the solution after an initial trajectory is identified, and asymptotically converge to an optimal solution. The RRT* method [5] provides an RRT with AO behavior by re-wiring the tree so that each node is connected to the roadmap root by a minimum-cost path. As the number of roadmap configurations increases, the cost of each path in the tree converges asymptotically towards the global optimum.

However, RRT* requires a steering function to support nonholonomic robots, and is thus not applicable to systems without a known steering function. To combat this difficulty, the Stable Sparse Tree or SST method provides an RRT-like algorithm which achieves asymptotic near-optimality behavior without a steering function [6]. The tree is extended only from the lowest-cost configuration within a given *witness region*, where the witness regions are small balls in state space that enforce sparseness. The SST* variant provides full AO behavior by slowly reducing the witness radius over time. Although tuning the sparseness is non-trivial, the SST is theoretically important in being the first algorithm to provide AO planning without a steering function.

C. Inner Distance

To support AO planners with the topological filter, we will require a measurement of the shortest-path distance between two cells in a workspace decomposition without entering the obstacle space. This is known as the inner distance [15] or Euclidean shortest-path distance. We will prefer the former term to avoid confusion with the general Euclidean distance. An exact computation of this distance is known to be NP-Hard [16], but bounded approximations exist which can estimate the distance in polynomial time [17].

III. Topological Filtering

Topological Filtering employs a convex cell decomposition of the workspace, which is a partitioning of the free workspace into a set of discrete convex cells [18], e.g. a tetrahedralization. The decomposition provides a graph representation of adjacent convex cells in the workspace and can be thought of as an atlas, which we refer to as the decomposition graph. This map encodes information about the connectivity of the cells and can be searched to locate sets of nearby cells, or *neighborhoods*.

The decomposition graph can also be used as a means of 'bucketing' nearby configurations together. A point on the robot's base, termed the *reference point*, is chosen to represent the robot's rough location in workspace. Usually the object's centroid or bounding box center is a good choice for this point. Let r be a robot, p be its reference point, and W be a decomposition graph. For any configuration $q \in \mathcal{C}_{free}$, a cell $c \in W$ is said to *contain* q if $p \in c$ when r is configured at q (Fig. 1(b)). The cell c which contains q may be efficiently determined by a range-searching method such as a segment tree [18]. Since the cells of W are disjoint, each free configuration will map to exactly one cell (boundary cases may be decided by any deterministic method).

The topological filter leverages this relationship by mapping configurations to their containing regions with a hash map. Whenever a configuration q is added to the roadmap, the filter maps q to the set for cell c, and conversely maps c to q. This *topological mapping* provides an amortized constant-time lookup of the vertices in a cell and the cell holding an already-discovered vertex (although the latter can be efficiently recomputed as noted above).

When searching for nearest neighbors to some configuration q, the filter first locates the cell c which contains q. Next, it performs a single-source shortest-path search through the decomposition graph W starting from q and looks for the

first cell c_1 that holds a mapped configuration. It continues searching until exceeding an additional *backtrack distance* and takes all cells discovered between c_1 and the end as the *topological frontier* F for this query. F is a set of cells which hold the most promising nodes for connection w.r.t. the subset of $\mathcal{C}_{space}$ that W is modeling (in this case, the physical workspace). The roadmap vertices in F can be determined using the topological map and used as the candidate set to draw neighbors from in a traditional nearest-neighbor method, e.g., a k-d tree. The filter's role is thus to select a small set of promising candidate neighbors.

This type of frontier selection represents a dynamically-selected set of cells that extends from the first populated cell to a backtrack distance further away from the query cell c, and it works well in combination with k-nearest neighbor selection strategies. For a radius-based nearest-neighbor method, we do not need to consider the state of the roadmap and simply select all cells within a radius of c.

IV. Distance Between Cells

Ideally, the distance between a pair of cells c_1, c_2 would be computed as the minimum inner distance (Section II-C) between any two points $p_1 \in c_1, p_2 \in c_2$, which is an NP-Hard problem. When approximate nearest neighbors are acceptable, we can employ a rough estimate of the inner distance with a decomposition graph search where the edge weights are the distance between cell centers, measured through the midpoint of the shared facet. While computationally cheap, it unfortunately has many failure cases and is not suitable for exact nearest neighbors.

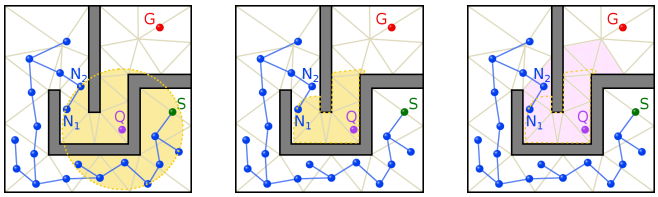
When we must have exact nearest neighbors (i.e. for asymptotically-optimal planning), we require a measure of the inner distance $T_{id}(c_1, c_2)$ between cells. We can accept an approximation $\hat{T}_{id}(c_1, c_2)$ so long as it is upper-bounded by a constant; i.e., $\hat{T}_{id}(c_1, c_2) \leq \delta T_{id}(c_1, c_2)$ for all c_1, c_2 and some $\delta > 0$. With such an approximation, the frontier F will be the set of cells where $\hat{T}_{id}(c_1, c_2) \leq \delta r_{\mathcal{W}}$ for an inner-distance $r_{\mathcal{W}}$.

One such approximation is to employ an occupancy grid search by overlaying a grid with voxel length s onto the workspace and mapping cells to voxels with collision checking. An outward Manhattan search over the grid from the voxels touching a cell c_1 can then determine the minimum distance to a voxel touching another cell c_2. This yields an approximation with $\delta = \sqrt{2}$ for two-dimensional workspaces and $\delta = \sqrt{3}$ for three-dimensional workspaces.

V. Topological Filtering with AO Planners

Asymptotically-optimal planners such as RRT* and SST require a radius-based search given the metric defined in $\mathcal{C}_{space}$ to identify candidate neighbors. In this section, we will refer to this radius as $r^* \geq r^*_{RRT}$, where r^*_{RRT} is ths minimum radius for asymptotic-optimality. To employ topological filtering with these methods, we require a radius-based frontier with a carefully chosen radius to capture the necessary portions of $\mathcal{C}_{space}$.

Preserving the optimality guarantees for either planner requires that the nearest-neighbor operation locate all configurations within r^* of a query configuration $q \in \mathcal{C}_{space}$. For a distance metric $D(q_1, q_2)$ which measures the proximity of two configurations q_1, q_2 through $\mathcal{C}_{space}$, this *radius criterion* indicates that all roadmap configurations $\{x \in \mathcal{C}_{free} \mid D(q, x) \leq r^*\}$ must be considered as candidates. However, this choice of radius is motivated by theoretical arguments of percolation theory in relation to random geometric graphs in spaces without obstacles [5]. In the presence of obstacles, we observe that percolation must occur in a graph over $\mathcal{C}_{free}$ rather than $\mathcal{C}_{space}$. Obstacle occlusion means that two configurations q_1, q_2 will only be connectable by a straight line if their inner distance in $\mathcal{C}_{free}$, denoted as $D_{id}(q_1, q_2)$, is equal to their distance ignoring obstacles, $D(q_1, q_2)$. This is because inner distance measures the length of the shortest path from q_1 to q_2 without leaving $\mathcal{C}_{free}$ (Fig. 2(a),2(b)), so $D(q_1, q_2) \neq D_{id}(q_1, q_2)$ indicates that the shortest path is not a straight line as would be attempted by the local planner. Consequently, percolation of the graph is maintained by substituting D_{id} for D when selecting neighbors because the former will never omit connectable candidates.



(a) (b) (c)

Fig. 2. (a) A $\mathcal{C}_{space}$ radius (yellow) as computed by a standard distance metric, projected onto workspace. (b) The same radius measured by inner distance. (c) The radius-based frontier (purple) selected to model the desired inner distance.

However, we cannot feasibly compute the set of configurations within a $\mathcal{C}_{space}$ ball β_{r^*} with respect to D_{id}. Our strategy then will be to leverage the workspace topology to approximate a projection of β_{r^*} into the workspace $\mathcal{W}$, where we can then leverage our topological map to locate a superset of the desired candidates. Formally, let β_{r^*} be a ball of radius r^* in the metric space $M_{\mathcal{C}} = (\mathcal{C}_{space}, D_{id})$ which contains some set of configurations $X_{\mathcal{C}} \subset \beta_{r^*}$. Let $\beta_{r_{\mathcal{W}}}$ be a ball in the metric space $M_{\mathcal{W}} = (\mathcal{W}, T_{id})$, where T_{id} represents the inner distance in $\mathcal{W}$ and $\beta_{r_{\mathcal{W}}}$ contains a set of configurations $X_{\mathcal{W}}$. We aim to compute an approximate mapping $\phi : M_C \to M_{\mathcal{W}}$ such that the co-image of a ball in $M_{\mathcal{W}}$ is a superset of its counterpart in $M_{\mathcal{C}}$, i.e. $\phi^{-1}(\beta_{r_{\mathcal{W}}}) \supset \beta_{\mathcal{C}}$. When this is possible, the topological filter is able to provide an over-estimate of the configurations in $\beta_{r_{\mathcal{W}}}$ which satisfy the radius criterion as measured through $\mathcal{C}_{free}$ by D_{id}.

To satisfy this requirement, we can leverage a relationship between the $\mathcal{C}_{space}$ metric $D_{id}(q_1, q_2)$ and the workspace metric $T_{id}(q_1, q_2)$. If $T_{id}(q_1, q_2) \leq \alpha D_{id}(q_1, q_2)$ for all $q_1, q_2 \in \mathcal{C}_{free}$ and some constant $\alpha > 0$, then all configurations within a D_{id}-distance of r^* are also within an T_{id}-distance of $r_{\mathcal{W}} = \alpha r^*$. In this case, we can determine the candidate neighbors which satisfy the radius criterion using a distance measurement in workspace. This relation between D_{id} and T_{id} holds for many popular distance metrics, including the canonical L^2 norm. For mobile-base robots, it can be satisfied

with $\alpha = 1$ since the L^2 norm will always be at least the translational displacement. This is an approximation of the ideal mapping between spaces ϕ that is sufficient to guarantee the desired co-image property.

For satisfying distance metrics, we can employ a topological frontier F which includes all cells within inner distance of αr^* from the cell c containing a query point q, where inner distance between cells c_1, c_2 indicates the minimum possible inner distance between any pair of points in each cell (Fig. 2(c)). This frontier describes a radius in connected workspace. We argue that this set F will contain all configurations $X_{\mathcal{W}}$, and therefore all configurations $X_{\mathcal{C}} \subset X_{\mathcal{W}}$ because the relationship between metric spaces establishes a valid approximation of the ideal mapping ϕ.

In simpler terms, the set F includes all configurations which may be connectable to q with a path through $\mathcal{C}_{free}$ of length r^* or less. The radius criterion in optimal planners is specifically meant to lower-bound this path distance, so our slight over-estimate will yield asymptotically-optimal behavior. Any configurations that are within an absolute D-distance of r^* to q with D_{id}-distance greater than r^* are necessarily occluded by an obstacle and will not be connectable to q.

VI. Extension to Manipulators

Applying the topological filter to multi-link robots presents an obvious question on whether a single reference point p_R is adequate to characterize the workspace neighborhood of the full robot body. A single p_R on the robot's base will certainly provide a valid filter for mobile-base manipulators, but this fails to capture connectivity for the remainder of the links. The same issue applies to fixed-base manipulators where the end-effector is the most important body. In these cases, there is no choice of p_R that provides a complete topological mapping for the entire robot.

The topological filter can be extended to support filtering on multiple links by generalizing the concept of a cell containing a configuration. Let $B = (b_1, b_2, \ldots, b_n)$ be an ordered tuple of the robot R's n individual component links. When R is positioned at some configuration q, each of its individual links $b_i \in B$ will be contained by some cell $c \in W$, where W is the decomposition graph. We define the *neighborhood key* of R at a configuration q as the ordered tuple of cells $(d_1, d_2, \ldots, d_n)$ occupied by R's individual links when positioned at q (Fig. 1(c)).

The set of all valid neighborhood keys and transitions between them is exceedingly large even for a small decomposition. Each link $b_i \in B$ may be contained by any cell $d_i \in W$, and multiple links may also be contained by the same cell such that $d_i = d_j$ for $i \neq j$. The number of possible neighborhood keys is thus $|W|^n$, so we unfortunately cannot form a graph over that space. However we can approximate the function of such a structure by composing a separate topological map for each of the robot's links.

To apply topological filtering to multiple links of R, we can construct a separate topological map for individual links. When searching for a nearest-neighbor to some configuration q, the filter now begins by locating the neighborhood key for q via the set of individual maps. The frontiers and candidates are then identified as in the rigid-body case and joined by a soft intersection to produce a refined candidate set.

To compute the soft intersection, we count the number of times each candidate appears across all frontiers and select the most frequently observed candidates. This identifies the best available candidates and empirically produces good results. It is achievable with the same time complexity as a strict intersection, which is undesirable because it may produce an empty candidate set. This can occur due to a relatively high mobility of the end-effector (and other distal links) in comparison with the base (and other proximal links). As the number of links and their lengths increase, this problem worsens because there are a greater variety of configurations that do not simultaneously occupy the topological frontiers for all links. An empty intersection is a worst-case scenario for the filter because the computation performed to locate candidates is wasted without discovering any useful information.

We also note that it is not strictly necessary to filter on all of the links, as there is a kinematic relationship between them defined by the properties of their adjacent joints. Depending on the mobility of the robot, it may be desirable to filter only one link or some subset of the links, such as the base, end-effector, and a middle elbow. The filter will ignore the unfiltered links in this case and only restrict the candidate set based on the links under consideration.

To preserve AO properties we require a radius-based frontier for each link. For fixed-base robots this is feasible by establishing a maximum ratio of workspace translation to $\mathcal{C}_{space}$ distance to define an α value for each link.

For mobile bases, this presents a challenge as there is often no meaningful way to bound the ratio for links other than the base. In such cases, a filter on the base alone will preserve the AO properties but lack the discerning power of a filter covering more links. The filter power can be increased by applying a filter with a carefully selected radius to other links, but this is a heuristic which unfortunately can't strictly preserve the AO properties of RRT* because a radius that is too small may miss connectable candidates (although the asymptotic near-optimality properties of SST will be preserved). Because this may rule out connectable configurations that are truly within the optimal radius r^*, the resulting planner may not converge to the true optimal path and is thus only near-optimal.

Despite the theoretical limitations, we observe good results in practice with mobile bases by applying the filtering radius for the base link to the end-effector only. This prefers to attempt rewiring between configurations with nearby end-effector positions at the cost of true asymptotic-optimality.

In some cases however, a workspace-based filter on the end-effector is counterproductive. This occurs for end-effector positions where the robot has high redundancy and thus self-mobility. High self-mobility means that there are many possible configurations that leave the base and end-effector fixed in the same relative configurations. It is not necessarily the case that transitioning between all of these configurations is easy or likely. For highly redundant manipulators, it is best to perform filtering on links that are more proximal to the base with lower self-mobility, or to at least include such a link

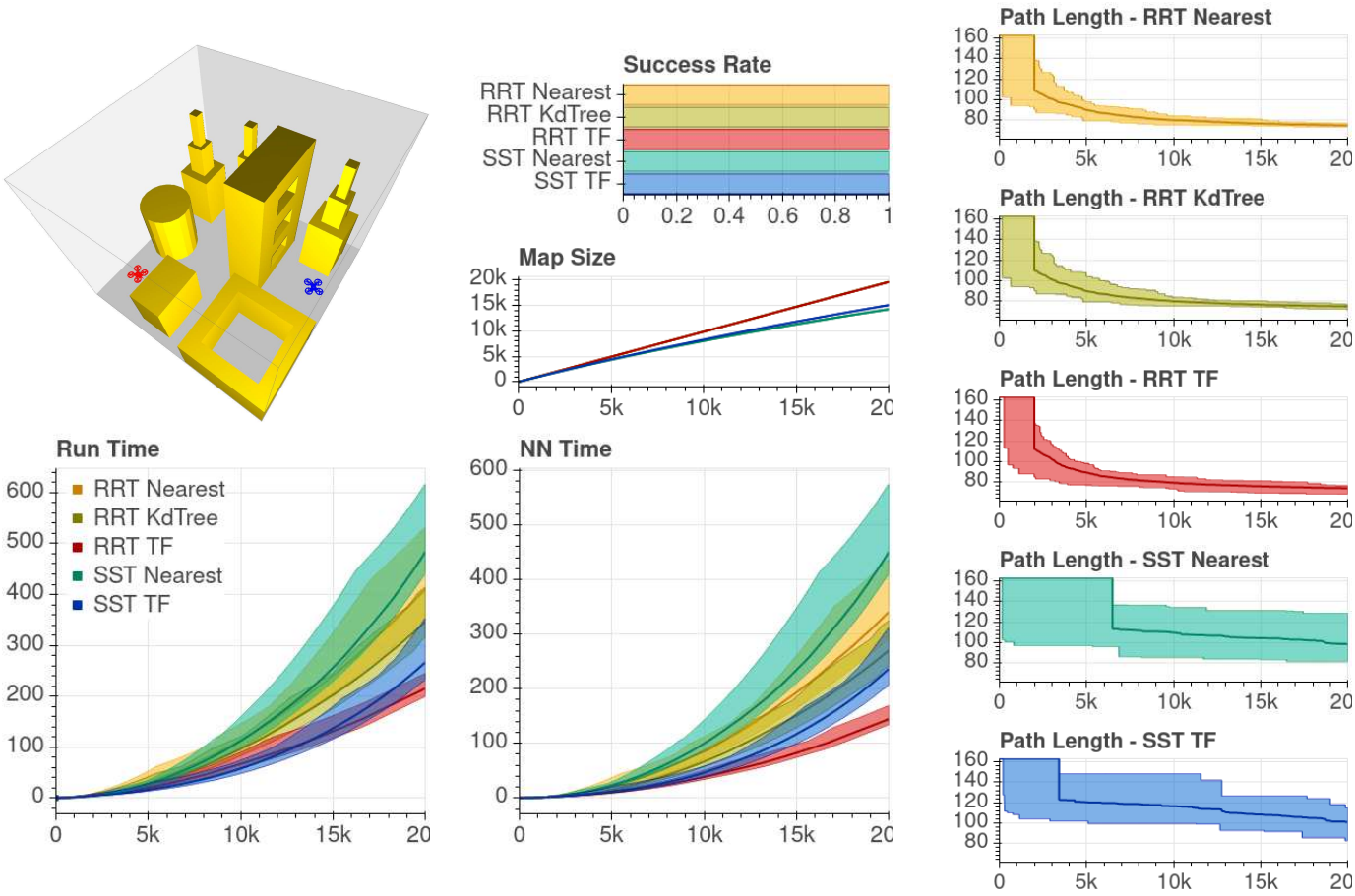


Fig. 3. A holonomic rigid-body problem where a quadcopter must traverse a cityscape. The solid lines show the average while the colored areas represent the spread.

in a composite multi-link filter to avoid this problem. This sacrifices some discriminating power but avoids over-fitting the neighbor selection on the assumption of easy transitions between configurations with nearby end-effector locations.

VII. Evaluation

To validate the method, we compare RRT* and SST with brute-force, k-d tree, and topological filtering over two 3D rigid-body (6 DOF) and two manipulator planning problems with thin walls that extend beyond feasibility planning and into path refinement. The k-d tree variant employs a k-d tree only during the 1-nearest check and not the radial searches, and is thus not applicable to SST.

A. Experiment Setup

The rigid-body problems include a quadcopter in an open environment (Fig. 3) and a box in a moderately narrow tunnel (Fig. 4). The first shows how the filter performs in more open spaces, where its primary advantage is reducing the number of candidate neighbors to consider. The second shows how its connectivity model provides additional benefits in workspaces with occluded visibility.

The manipulator problems include a fixed-base manipulator maneuvering between shelves (Fig. 5) and a mobile-base manipulator moving around shelves (Fig. 6). This covers a range of manipulator problems where the topological filter is expected to provide some level of utility by avoiding candidate neighbors likely to intersect with thin walls. In all cases, the filtering is performed only on the end-effector to maximize this potential benefit (which also implies that the RRT* variants are limited to asymptotic near-optimality).

The same robot is used for the fixed-base and mobile-base problems. It has four spherical joints and is permitted to translate but not rotate its base in the mobile version (because rotation would be redundant with the first joint). It has eight DOF for the fixed problem and ten for the mobile one.

All methods employ a $\mathcal{C}_{space}$ Euclidean distance metric with straight-line local planning and uniform random sampling in $\mathcal{C}_{free}$. Each variation is run thirty times to 20k iterations. We report the generated map size, run time, nearest-neighbor time, path cost, and success rate (where success is defined as generating a path before the iteration limit). The plots show an average as well as the minimum/maximum envelope for each method as a function of iteration count. The envelope shows how the methods overlap in expected behavior. The use of iterations rather than time for the x-axes is intended to disambiguate the effects of cheaper vs. more efficacious iterations. Times are reported in seconds and path cost in Euclidean distance. Only successful trials are plotted.

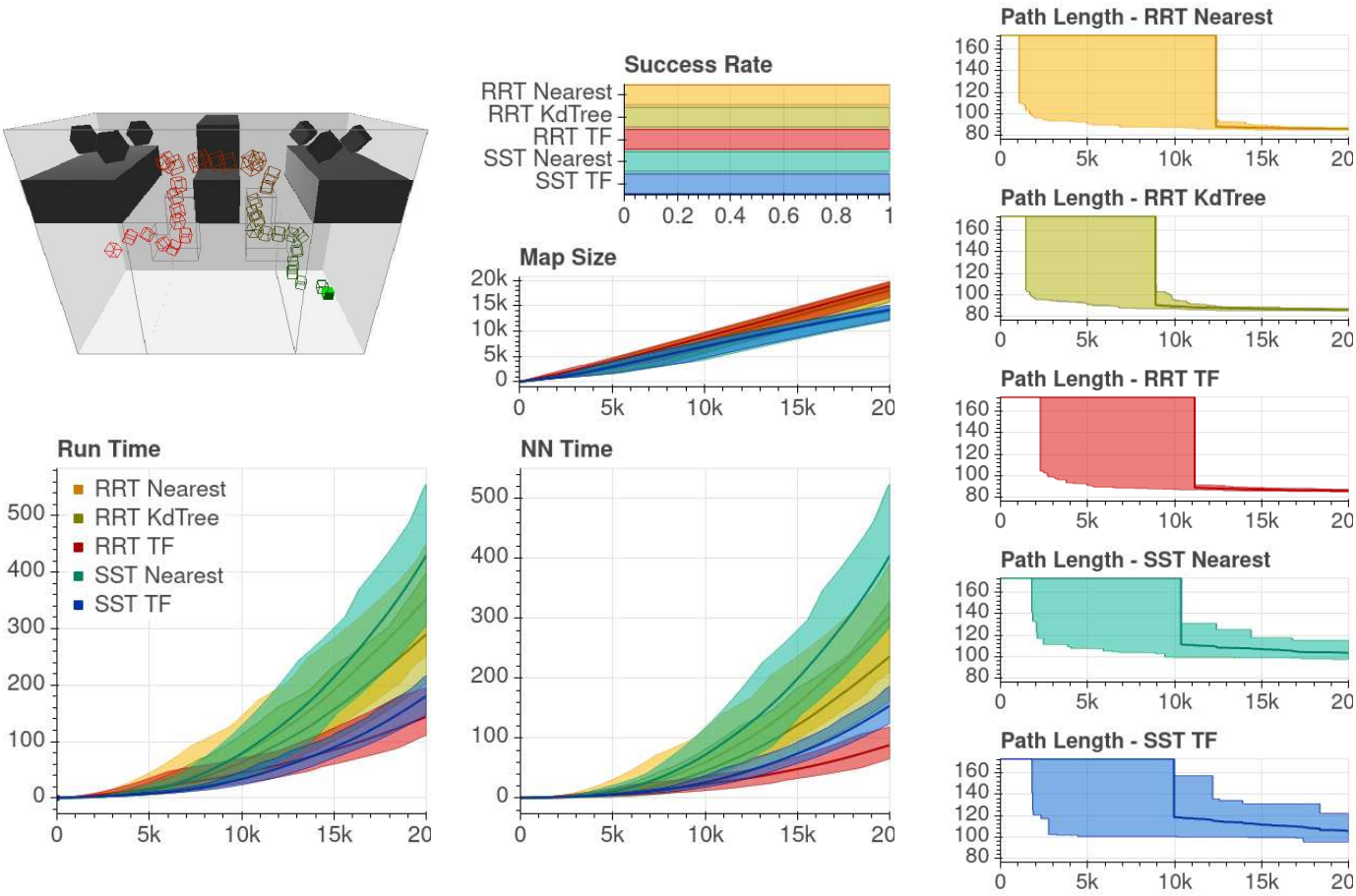


Fig. 4. A holonomic rigid-body problem where a box robot must traverse two L-shaped tunnels and a narrow gap. The rotation of the box substantially increases the narrowness of the tunnels in $\mathcal{C}_{space}$. The solid lines show the average while the colored areas represent the spread.

All experiments were executed on a desktop computer running CentOS 7 with an Intel® Core™ i7-3770 CPU at 3.4 GHz, 16 GB of RAM, and the GNU g++ compiler version 9.2.0. The workspace tetrahedralization was performed with a combination of the TetGen [19] and CGAL [20] libraries and required no more than one second to complete. The k-d tree implementation is also from the CGAL library.

B. Analysis

1) Rigid-body Problems: In the quadcopter problem (Fig. 3), we see that the filter provides consistently faster run and nearest-neighbor time for both methods vs. the brute-force and k-d tree versions. The gain is sufficient that the envelope maximum for the filtered algorithms performs better than the minimum for the non-filtered instances. We also observe no loss in path quality, and that the filtered algorithms take equal or less time to discover an initial path.

In the tunnel problem (Fig. 4), we observe a similar trend: the filtered algorithms' maximum envelope lies beneath the unfiltered variants' for both time metrics. There is no loss in average path quality for either RRT* or SST, although the filtered SST exhibits slightly higher variance than the unfiltered version. The number of iterations until all paths solve is also slightly higher for the filtered RRT* in comparison to the k-d tree version, but the filter's advantage in run time means that it will reach that point before the k-d tree would.

In both problems, we see a clear advantage in both run-time and nearest-neighbor time for the filtered methods which continues to increase as the iteration count grows. This supports the hypothesis that the filter's efficiency and efficacy grow with roadmap size, which is a very desirable property.

2) Manipulator Problems: In the fixed-base problem (Fig. 5), we observe reasonable benefits from the filter in all metrics of interest. The execution and nearest-neighbor time envelopes for the filtered RRT* are clearly below the unfiltered versions, and the path cost converges more quickly to a lower minimum value. The SST variants struggle with this problem because the notion of a witness radius does not work well with fixed-base manipulators. The problem here is that the robot's $\mathcal{C}_{space}$ is entirely comprised of its joint space, and small changes in joint space values can yield drastically different changes in the end-effector position. This means that small motions of the robot may be rejected due to a lower-cost witness 'nearby' even though the configurations are significantly different. The witness regions are meant to define small neighborhoods in $\mathcal{C}_{free}$ where the contained configurations have similar connectivity properties, but this breaks down for joint space due to the large differences in semantics that can occur with small

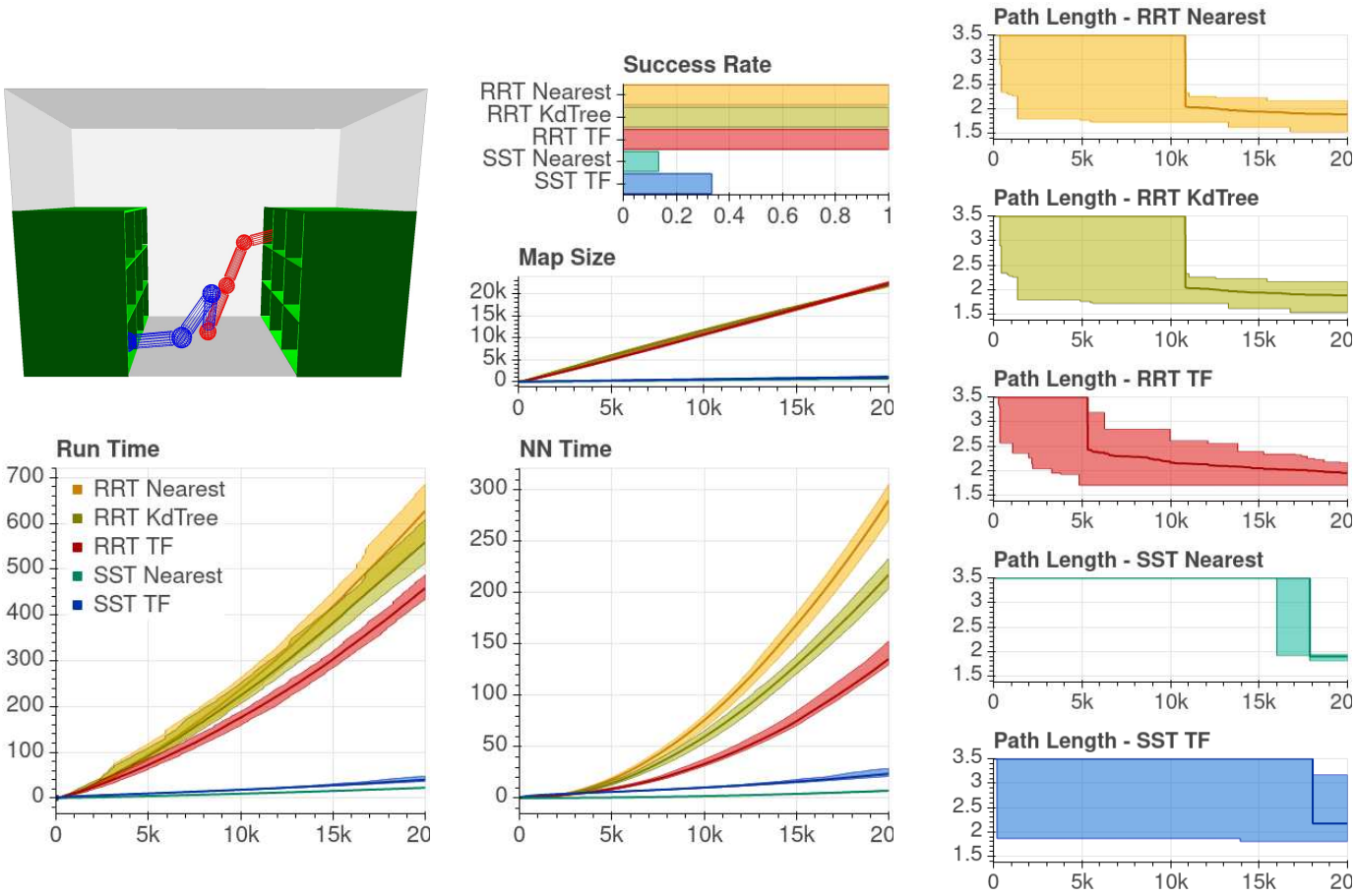


Fig. 5. A fixed-base manipulator problem where the robot must transition between grasping positions within pick shelves on alternate sides of the workspace. The solid lines show the average while the colored areas represent the spread.

differences in the $\mathcal{C}_{space}$ metric.

In the mobile-base problem (Fig. 6), we observe that the SST problems are mitigated by the influence of the translational DOFs on the distance metric. However, the filtered SST algorithm under-performs compared to the unfiltered version in convergence rate; it does achieve equivalent path cost but takes longer to do so, with more trials failing to discover a path by the iteration limit. This occurs despite a better extension success rate indicated by the larger map size for the filtered version. For RRT*, we also observe a higher rate of successful extension and comparable path cost, but with quicker discovery of initial solution. The filtered algorithm's nearest-neighbor time is roughly comparable to the k-d tree version despite working with a 25% larger roadmap.

These problems demonstrate that the filtering concept can be applied to asymptotically-optimal manipulator problems. Generally we observe that the step down to asymptotic near-optimality with RRT* by filtering on the end-effector for mobile base problems provides a better convergence rate to lower path cost despite the reduced guarantee. This occurs because filtering during the extension step changes the shape of the tree to avoid excessively driving configurations up against obstacle walls, causing more of the extensions to be again extendable in a subsequent iteration. This also causes the higher rate of extension success observed by the larger maps generated by the filter. Filtering during the rewiring step helps manage the cost of rewiring the larger map, and avoids costly local plans which are unlikely to succeed.

VIII. Conclusion

We describe the topological filtering algorithm for nearest-neighbor search with asymptotically-optimal planners, and demonstrate that it both improves the likelihood of successful extension and reduces the computational cost of the nearest-neighbor process. These benefits arise from a level of obstacle-awareness in nearest-neighbor selection provided by a an approximation of $\mathcal{C}_{free}$'s connectivity.

An interesting avenue for future work is to apply the concepts here with an atlas other than a workspace decomposition, such as those generated by the AtlasRRT algorithm [21]. Because the filter only requires that the cells or charts represent neighborhoods where connectivity is likely, virtually any atlas with such a property could be employed. An approximate atlas may allow application of the filtering technique to problems where the salient details of valid paths in $\mathcal{C}_{free}$ bear little or no relation to workspace, such as the well-known alpha puzzle.

References

[1] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration

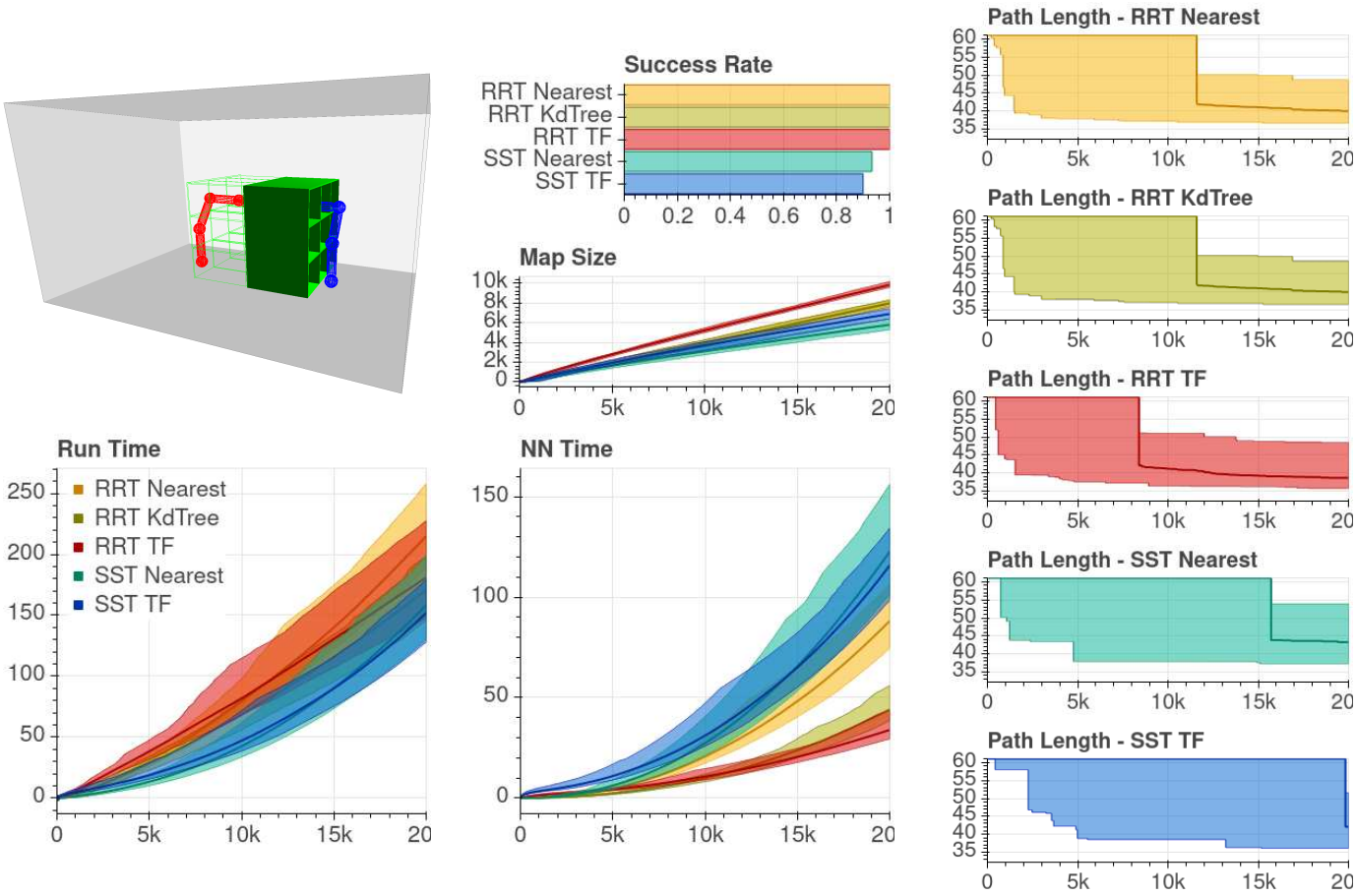


Fig. 6. A mobile-base manipulator problem where the robot must transition between grasping positions within back-to-back pick shelves by translating around the exterior. The reverse shelf is shown in wire-frame for visual clarity of the goal position. The solid lines show the average while the colored areas represent the spread.

spaces," *IEEE Trans. Robot. Automat.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[2] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1999, pp. 473–479.

[3] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Softw.*, vol. 3, pp. 209–226, 1977.

[4] R. Sandström, A. Bregger, B. Smith, S. Thomas, and N. M. Amato, "Topological nearest-neighbor filtering for sampling-based planners," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018, pp. 3053–3060.

[5] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, pp. 846–894, 2011.

[6] Y. Li, Z. Littlefield, and K. E. Bekris, "Sparse methods for efficient asymptotically optimal kinodynamic planning," in *Alg. Found. Robot. XI*. Springer, 2015, pp. 263–282, (WAFR '14). [Online]. Available: http://dx.doi.org/10.1007/978-3-319-16595-0_16

[7] R. Sandström, "Approximating configuration space topology with workspace models," Ph.D. dissertation, Texas A&M University, 2020.

[8] A. Yershova and S. M. LaValle, "Improving motion-planning algorithms by efficient nearest-neighbor searching," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 151–157, 2007.

[9] J. Ichnowski and R. Alterovitz, "Fast nearest neighbor search in SE(3) for sampling-based motion planning," in *Alg. Found. Robot. XI*. Springer, 2015, pp. 197–213, (WAFR '14).

[10] A. M. Kibriya and E. Frank, "An empirical comparison of exact nearest neighbor algorithms," in *Proc. of the Euro. Conf. Data Min. Know. Disc.*, 2007.

[11] J. Pan, C. Lauterbach, and D. Manocha, "Efficient nearest-neighbor computation for GPU-based motion planning," in *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, 2010.

[12] P. Indyk, R. Motwani, P. Raghavan, and S. Vempala, "Locality-preserving hashing in multidimensional spaces," in *Proc. Annu. ACM Sympos. Theory Comput.*, 1997, pp. 618–625.

[13] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirronki, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proc. Annu. Symp. on Comput. Geom.*, 2004, pp. 253–262.

[14] E. Plaku and L. Kavraki, "Quantitative analysis of nearest-neighbors search in high-dimensional sampling-based motion planning," in *Alg. Found. Robot. VII*. Springer, 2006, pp. 3–18, (WAFR '06).

[15] H. Ling and D. W. Jacobs, "Shape classification using the inner-distance," *IEEE Trans. Patt. Analy. Mach. Intell.*, vol. 29, no. 2, pp. 286–299, 2007.

[16] J. Canny and J. Reif, "New lower bound techniques for robot motion planning problems," in *Proc. IEEE Symp. Found. Comp. Sci. (FOCS)*. IEEE, 1987, pp. 49–60.

[17] Y.-S. Liu, K. Ramani, and M. Liu, "Computing the inner distances of volumetric models for articulated shape description with a visibility graph," *IEEE Trans. Patt. Analy. Mach. Intell.*, vol. 33, no. 12, pp. 2538–2544, 2011.

[18] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*, 2nd ed. Berlin, Germany: Springer-Verlag, 2000.

[19] H. Si, "TetGen, a delaunay-based quality tetrahedral mesh generator," *ACM Trans. Math. Softw.*, vol. 41, no. 2, pp. 11:1–11:36, Feb. 2015. [Online]. Available: http://doi.acm.org/10.1145/2629697

[20] A. Fabri, G.-J. Giezeman, L. Kettner, S. Schirra, and S. Schönherr, "On the design of CGAL a computational geometry algorithms library," *Softw. – Pract. Exp.*, vol. 30, no. 11, pp. 1167–1202, 2000.

[21] L. Jaillet and J. M. Porta, "Path planning under kinematic constraints by rapidly exploring manifolds," *IEEE Trans. Robot.*, vol. 29, no. 1, pp. 105–117, 2013.