Sensory Steering for Sampling-Based Motion Planning

Omur Arslan and Vincent Pacelli and Daniel E. Koditschek

***Abstract*— Sampling-based algorithms offer computationally efficient, practical solutions to the path finding problem in high-dimensional complex configuration spaces by approximately capturing the connectivity of the underlying space through a (dense) collection of sample configurations joined by simple local planners. In this paper, we address a long-standing bottleneck associated with the difficulty of finding paths through narrow passages. Whereas most prior work considers the narrow passage problem as a sampling issue (and the literature abounds with heuristic sampling strategies) very little attention has been paid to the design of new effective local planners. Here, we propose a novel sensory steering algorithm for sampling-based motion planning that can "feel" a configuration space locally and significantly improve the path planning performance near difficult regions such as narrow passages. We provide computational evidence for the effectiveness of the proposed local planner through a variety of simulations which suggest that our proposed sensory steering algorithm outperforms the standard straight-line planner by significantly increasing the connectivity of random motion planning graphs.**

I. Introduction

The modern use of robots, such as in household applications [1], package delivery [2], warehouse management [3], and transportation [4], requires finding safe navigation paths in complex-shaped, high-dimensional configuration spaces that are generally very difficult, if not impossible, to represent explicitly. Fortunately, sampling-based motion planning methods (e.g., probabilistic roadmaps [5], rapidly-exploring random trees [6], and their variants) offer computationally efficient solutions to path planning in such complicated configuration spaces by approximately modeling their connectivity using a (dense) collection of sample configurations that are connected by simple local planners. As one might expect, it is usually a challenging task to identify narrow passages in configuration spaces and find a path through such limited regions, especially using randomized approaches [7], [8]. In this paper, we introduce a new *sensory steering* algorithm for sampling-based motion planners that can "feel" the local geometric structure of a configuration space around a sample configuration and can generate effective steering motion near narrow passages, as illustrated in Fig. 1.

A. Motivation and Prior Literature

Although the use of (asymptotically dense) sample configurations connected by simple local planners enables computationally affordable, approximate modeling of configuration spaces with probabilistic completeness guarantees, sampling-based motion planners, with no additional special treatment, are known to perform less well around narrow passages [7], [8]. For instance, uniform sampling is known to have a Voronoi bias causing rapid exploration of wider regions of configuration spaces (i.e., reduced sampling from narrow passages) [9]; similarly, the standard straight-line planner is very limited in capturing local connectivity near narrow passages, because it only connects a pair of sample configurations if the straight line joining them is free of collisions.

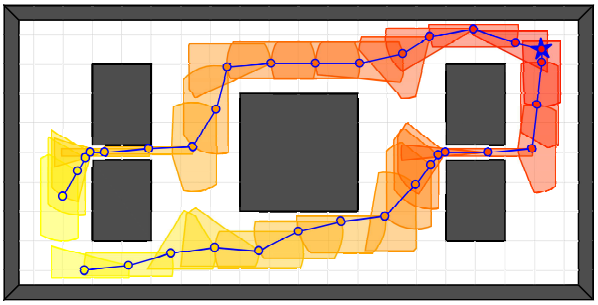


Fig. 1. Sensory steering steps (blue line segments) toward a goal (blue star-shaped marker) among obstacles (dark gray polygons). Using critical collisions around a sample configuration, sensory steering first identifies an obstacle free convex neighborhood of the configuration, and then chooses a step toward the closest point of this safe zone to the goal. By construction, sensory steering is scale robust and preliminary numerical evidence (e.g., see Figs. 4 & 5) suggests that it substantially outperforms the standard straight-line planner by significantly increasing connectivity of sample configurations, especially near narrow passages.

The majority of past work has treated the narrow passage problem as a sampling issue. Accordingly, many heuristic strategies that bias sampling towards narrow passages have been proposed based on geometric properties of configuration spaces and sampling history. Representative approaches include: retraction onto the medial-axis [10]–[12] and the boundary [13] of the free space[1]; cell decomposition based sampling [14]–[17]; bridge-test sampling [18]; Gaussian sampling [19]; entropy based sampling [20]; artificial potential biased sampling [21]; human-guided sampling [22]; simultanous sampling of the free space and configuration space obstacles [23]; sampling using collision information [5], [24]; and their combinations [25].

Alternatively, the difficulty of path finding around narrow passages can be mitigated by designing effective local planners. Sampling-based [26], [27] and search-based (e.g., A*-like) [13], [28], [29] local planning approaches have been demonstrated to perform better than the standard straight-line planner near narrow passages, but these approaches require storage of path segments joining sample configurations and so cause an increase in memory requirements. This additional

The authors are with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104. E-mail: {omur, pacelliv, kod}@seas.upenn.edu. This work was supported in part by AFRL grant FA865015D1845 (subcontract 669737-1).

[1] We use the terms "free space" and "configuration space" interchangeably. It also bears mentioning that, with no approximation, retraction onto the medial-axis or the boundary of the free space and cell decomposition methods require an explicit representation of configuration spaces.

memory complexity can be reduced by constraining the search space to a low dimensional subspace and/or limiting the number of vertices used to represent path segments [30]. Randomized potential field methods [31] that combine the desired strengths of artificial potential functions [32] and sampling-based methods, are successfully applied to the path finding problem, but they inherently suffer from getting trapped around local minima [33], because escape routes from such traps might be narrow.

B. Contributions and Organization of the Paper

As an alternative local planner, in this paper we propose a novel local path planning algorithm for sampling-based motion planning that tries to connect sample configurations by exploiting, in a computationally effective manner, "sensed" local geometric structure of configuration spaces around sample configurations. More precisely, using the critical points of the free space boundary around a sample configuration, computed from a "sensor" that returns the (range limited) closest points of convex obstacles, we first construct a convex collision-free neighborhood of the sample configuration, and then accordingly build our local planner that steers the sample configuration toward the closest point of its convex local safe neighborhood to any given sample destination configuration. We show that an incremental anytime version of the proposed local planner can be implemented efficiently using active set methods of convex quadratic optimization [34]. We demonstrate in simulations the effectiveness of the proposed local planner for finding paths through narrow passages, exploring how well it captures the local connectivity of configuration spaces around such difficult regions. It is worth mention that this construction is an adaptation of our recently introduced feedback motion planner for collision free global navigation in densely cluttered environments with convex obstacles [35], [36]. Whereas the assumptions underlying the formal guarantees associated with those methods cannot be expected to hold in the general application settings addressed here, their obstacle-avoiding, convergence-seeking "greedy" design may help intuitively explain why the proposed sensory steering algorithm performs significantly better than the straight-line planner near narrow passages.

This paper is organized as follows. Section II briefly reviews background on sampling-based motion planning. Section III, comprising the central contribution of the paper, constructs a convex local free space around a sample configuration and presents our sensory steering algorithm. Section IV provides a brief discussion on computational complexity and implementation suggestions. Section V illustrates the effectiveness of the proposed planner using numerical simulations. Section VI concludes with a summary of our contributions and a brief discussion of future directions.

II. Background: Sampling-Based Motion Planning

In this section, we briefly review the generic components of sampling-based motion planning algorithms, and present two widely used randomized motion planners, probabilistic roadmaps (PRMs) [5] and rapidly-exploring random trees (RRTs) [6]. First, it is convenient to introduce some notation. For a robotic system, let $\mathcal{F} \subseteq \mathbb{R}^n$ denote its compact closed free space that contains the set of robot configurations with no collisions, for some $n \in \mathbb{N}$; and let $[\mathrm{x}, \mathrm{y}] := \left\{ \alpha \mathrm{x} + (1-\alpha)\, \mathrm{y} \in \mathbb{R}^n \middle| 0 \leq \alpha \leq 1 \right\}$ be the straight line segment between two points $\mathrm{x}, \mathrm{y} \in \mathbb{R}^n$.

A. Generic Elements

A sampling-based motion planning algorithm typically consists of the following generic components.

1) Sampling: A sampling method, denoted by $\texttt{Sample}(\mathcal{F})$, generates independent and identically distributed random[2] configurations from the free space $\mathcal{F}$, with a possible bias toward important regions such as narrow passages and the boundary of the free space. For example, one might consider any heuristic sampling approach referred to in Section I-A. In this paper, to highlight the strength of the proposed local planner, we generate uniform samples from the free space using rejection sampling that repeatedly draws sample points from a box-shaped subset of $\mathbb{R}^n$ containing $\mathcal{F}$ until finding a collision free configuration in $\mathcal{F}$.

2) Distance Measure: A distance function quantifies the relative proximity of a pair of sample configurations in the free space. Hence, an informative distance measure should be ideally as close as possible to the geodesic distance so that it expedites exploration of the free space. In this paper, we simply consider the standard Euclidean metric, denoted by $\|.\|$, and refer to [28], [37] for other alternative distance metrics.

3) Nearest Neighbor Search: For a choice of a distance function, nearest neighbor search, denoted by $\texttt{NearestNeighbor}(\mathcal{X}, \mathrm{x})$, seeks for the closest element of a collection of sample configurations $\mathcal{X} = \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_m\} \subset \mathcal{F}$ to a newly generated sample configuration $\mathrm{x} \in \mathcal{F}$. Typically, the underlying assumption is that the closest pairs of sample configurations are more likely to be connected by a simple local planner with no collisions. For the standard Euclidean metric, nearest neighbor search can be performed in logarithmic time using efficient data structures such as kd-trees [38].

4) Local Planner: A local planner, also known as a steering function, denoted by $\mathrm{z} = \texttt{Steer}(\mathrm{x}, \mathrm{y})$, suggests a greedy motion step from a sample configuration $\mathrm{x} \in \mathcal{F}$ toward a sample destination configuration $\mathrm{y} \in \mathcal{F}$ via an intermediate point $\mathrm{z} \in \mathbb{R}^n$ that is "closer" to y than x, with a possible upper bound $\epsilon > 0$ on the step size, i.e., $\|\mathrm{z} - \mathrm{x}\| \leq \epsilon$. A deterministic local planner is always preferred over a randomized local planner, because a deterministic planner does not require storage of path segments joining sample configurations [28]. A widely used deterministic local planner is the straight-line planner, defined as

$$\texttt{StraightSteer}(\mathrm{x}, \mathrm{y}) := \underset{\mathrm{z} \in B(\mathrm{x}, \epsilon)}{\arg\min} \|\mathrm{z} - \mathrm{y}\|, \qquad (1)$$

where $B(\mathrm{x}, \epsilon) := \left\{ \mathrm{p} \in \mathbb{R}^n \middle| \, \|\mathrm{p} - \mathrm{x}\| \leq \epsilon \right\}$ denotes the closed Euclidean ball centered at x with radius ϵ.

[2] A sampling method might also be deterministically constructed based on lattice-like regular structures [8].

To ensure the probabilistic completeness of a sampling-based motion planner, one can consider a local planner that can always join certain nearby configurations [39]:

Definition 1 A steering function $\texttt{Steer}: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ is said to be *admissible* if for any $\mathrm{x} \in \mathcal{F}$,

$$\texttt{Steer}\left(\mathrm{x},\mathrm{y}\right) = \mathrm{y}, \quad \forall \mathrm{y} \in B\left(\mathrm{x}, \tfrac{r}{2}\right), \tag{2}$$

where $r := \min(2\epsilon, d(\mathrm{x},\partial\mathcal{F}))$ and $d(\mathrm{x},\partial\mathcal{F}) := \arg\min_{\mathrm{z}\in\partial\mathcal{F}} \|\mathrm{x}-\mathrm{z}\|$ is the distance of x to the boundary $\partial\mathcal{F}$ of the free space $\mathcal{F}$.

For example, the standard straight-line planner is known to be admissible, since for any $\mathrm{x},\mathrm{y} \in \mathbb{R}^n$ with $\|\mathrm{x}-\mathrm{y}\| \leq \frac{r}{2}$ one has $B\left(\mathrm{y}, \frac{r}{2}\right) \subseteq B\left(\mathrm{x}, r\right)$ [39]. In this paper, we introduce a new deterministic, admissible sensory steering algorithm that can be used with any existing sampling-based motion planner while preserving probabilistic completeness guarantees.

Finally, to enable taking more than one concatenated motion steps for joining sample configurations, we find it convenient to define K-step motion steering as: for $K \geq 0$,

$$\texttt{Steer}^{K+1}(\mathrm{x},\mathrm{y}) := \texttt{Steer}\big(\texttt{Steer}^{K}(\mathrm{x},\mathrm{y}),\mathrm{y}\big), \tag{3}$$

where we set $\texttt{Steer}^{0}(\mathrm{x},\mathrm{y}) = \mathrm{x}$. Note that if Steer is deterministic and admissible, then so is $\texttt{Steer}^{K}$, for $K \geq 1$.

5) Collision Detection: A collision detector, denoted by $\texttt{CollisionFree}(\mathrm{x},\mathrm{y})$, checks if the straight line connecting a pair of sample points, $\mathrm{x},\mathrm{y} \in \mathbb{R}^n$, is free of collisions; and it returns *true* if $[\mathrm{x},\mathrm{y}] \subseteq \mathcal{F}$, and *false* otherwise. One can accurately determine collisions using fast incremental distance computation between convex polyhedra [40], or it can be approximately computed using binary search along a discretized line segment joining sample configurations. In this paper, to incrementally compute distance between convex polytopes, we use active set methods of convex optimization [34], briefly presented in Section IV-B, because they offer a natural generalization of [40] to arbitrary space dimensions. Further, abusing the notation, we shall check the safety of a steering step by $\texttt{CollisionFree}(\texttt{Steer}(\mathrm{x},\mathrm{y})) := \texttt{CollisionFree}(\mathrm{x},\texttt{Steer}(\mathrm{x},\mathrm{y}))$, and, likewise, the safety of K-step steering motion can be determined by

$$\texttt{CollisionFree}\Big(\texttt{Steer}^{K+1}(\mathrm{x},\mathrm{y})\Big) := \bigwedge_{k=0}^{K} \texttt{CollisionFree}\Big(\texttt{Steer}^{k}(\mathrm{x},\mathrm{y}), \texttt{Steer}^{k+1}(\mathrm{x},\mathrm{y})\Big). \tag{4}$$

B. Generic Algorithms

We now present basic versions (in Algorithm 1 and Algorithm 2) of the two widely used random motion planning graphs, probabilistic roadmaps (PRMs) [5] and rapidly-exploring random trees (RRTs) [6] for multi-query and single-query path planning applications, respectively.

A probabilistic roadmap $G = (V, E)$ consists of a finite collection of sample configurations as its vertex set V, and a pair of vertices $\mathrm{v} \neq \mathrm{u} \in V$ are connected by an undirected edge in E if and only if they can be safely joined together in K steering steps[3], i.e.,

$$\begin{matrix}(\mathrm{v},\mathrm{u})\\(\mathrm{u},\mathrm{v})\end{matrix} \in E \Leftrightarrow \begin{matrix}\big(\texttt{Steer}^{K}(\mathrm{v},\mathrm{u}) = \mathrm{u}\big) \wedge \texttt{CollisionFree}\big(\texttt{Steer}^{K}(\mathrm{v},\mathrm{u})\big)\\ \text{or}\\ \big(\texttt{Steer}^{K}(\mathrm{u},\mathrm{v}) = \mathrm{v}\big) \wedge \texttt{CollisionFree}\big(\texttt{Steer}^{K}(\mathrm{u},\mathrm{v})\big)\end{matrix} \tag{5}$$

Algorithm 1 shows how to construct such a PRM. Hence, after the construction phase, in the query phase, one can find a navigation path between a start and a goal configuration by first safely connecting them to the constructed PRM, and then searching a (shortest) path of the PRM between the associated terminal nodes.

Algorithm 1: Probabilistic Roadmap (PRM) [5]

Input: N – Number of Samples
K – Number of Steering Steps
Output: $G = (V, E)$ – Random Motion Planning Graph

$V \leftarrow \bigcup_{i=1}^{N} \{\texttt{Sample}\,(\mathcal{F})\}$; $E \leftarrow \varnothing$;
foreach $\mathrm{v} \neq \mathrm{u} \in V$ **do**
  if $\big(\texttt{Steer}^{K}(\mathrm{v},\mathrm{u}) = \mathrm{u}\big) \wedge \big(\texttt{CollisionFree}\big(\texttt{Steer}^{K}(\mathrm{v},\mathrm{u})\big)\big)$
    or
    $\big(\texttt{Steer}^{K}(\mathrm{u},\mathrm{v}) = \mathrm{v}\big) \wedge \big(\texttt{CollisionFree}\big(\texttt{Steer}^{K}(\mathrm{u},\mathrm{v})\big)\big)$
  then
    $E \leftarrow E \cup \{(\mathrm{v},\mathrm{u}),(\mathrm{u},\mathrm{v})\}$;
return $G = (V, E)$;

A rapidly-exploring random tree is an incrementally constructed motion planning graph $G = (V, E)$ such that its construction is initiated at a start configuration $\mathrm{x}_{\texttt{start}} \in \mathcal{F}$ and it is iteratively expanded toward a random sample $\mathrm{x}_{\texttt{rand}} = \texttt{Sample}\,(\mathcal{F})$ from its closest vertex $\mathrm{x}_{\texttt{nearest}} = \texttt{NearestNeighbor}\,(V, \mathrm{x}_{\texttt{rand}})$ using a safe steering step to $\mathrm{x}_{\texttt{new}} = \texttt{Steer}\,(\mathrm{x}_{\texttt{nearest}}, \mathrm{x}_{\texttt{rand}})$, as illustrated in Algorithm 2. Thus, to reach a goal set, one can expand an RRT until it contains a vertex from the goal set.

Algorithm 2: Rapidly-Exploring Random Tree (RRT) [6]

Input: N – Number of Iterations
$\mathrm{x}_{\texttt{start}} \in \mathcal{F}$ – Start Configuration
Output: $G = (V, E)$ – Random Motion Planning Graph

$V \leftarrow \mathrm{x}_{\texttt{start}}$; $E \leftarrow \varnothing$;
for $i = 1, \ldots, N$ **do**
  $\mathrm{x}_{\texttt{rand}} \leftarrow \texttt{Sample}\,(\mathcal{F})$;
  $\mathrm{x}_{\texttt{nearest}} \leftarrow \texttt{NearestNeighbor}\,(V, \mathrm{x}_{\texttt{rand}})$;
  $\mathrm{x}_{\texttt{new}} \leftarrow \texttt{Steer}\,(\mathrm{x}_{\texttt{nearest}}, \mathrm{x}_{\texttt{rand}})$;
  if $\texttt{CollisionFree}\,(\mathrm{x}_{\texttt{nearest}}, \mathrm{x}_{\texttt{rand}})$ **then**
    $V \leftarrow V \cup \{\mathrm{x}_{\texttt{new}}\}$; $E \leftarrow E \cup \{(\mathrm{x}_{\texttt{nearest}}, \mathrm{x}_{\texttt{new}})\}$;
return $(G = (V, E))$;

III. Sensory Steering

We now introduce a new greedy deterministic admissible sensory steering function for sampling-based motion planning, whose construction exploits the local geometric structure of a configuration space by identifying a convex collision-free neighborhood of a sample configuration using the critical collisions around it.

[3]Although the simplified PRM in [39] assumes a limited-range straight-line connectivity, we here find it convenient to construct a PRM using an arbitrary steering function and an arbitrary fixed number of steering steps.

A. Safe Neighborhood of a Sample Configuration

1) Local Free Space from Critical Collision Sensing: For ease of exposition, in this part we assume the a priori unknown configuration space obstacles can be accurately approximated as a finite union of closed convex polytopes[4], say $\{Q_1, Q_2, \ldots, Q_m\}$, where $m \in \mathbb{N}$; and we further assume the availability of a computational "sensing" model, denoted by $\mathcal{S}(\mathrm{x})$, that returns the closest points of convex obstacles, within a certain fixed sensing range $R > 0$, to a sample configuration $\mathrm{x} \in \mathcal{F}$, i.e.,

$$\mathcal{S}(\mathrm{x}) := \left\{ \Pi_{Q_i}(\mathrm{x}) \,\middle|\, d(\mathrm{x}, Q_i) \leq R,\, i \in \{1, \ldots, m\} \right\}, \quad (6)$$

where $d(\mathrm{x}, Q_i) := \min_{\mathrm{q} \in Q_i} \|\mathrm{x} - \mathrm{q}\|$ is the distance between point x and obstacle Q_i, and $\Pi_{Q_i}(\mathrm{x}) := \arg\min_{\mathrm{q} \in Q_i} \|\mathrm{x} - \mathrm{q}\|$ denotes the metric projection of x onto Q_i and returns the unique closest point of Q_i to x [41]. Metric projection onto a convex polytope can be recast as a linearly constrained least squares (i.e., convex quadratic optimization) problem and so can be computed in polynomial time, for example, using the active set method of convex quadratic optimization [34], summarized in Section IV-B. We refer to this local sensing model as *limited range critical collision sensing*.

Accordingly, given critical collision points $\mathcal{S}(\mathrm{x})$ around a sample configuration $\mathrm{x} \in \mathcal{F}$, using Voronoi decomposition, we construct the *local free space* $\mathcal{LF}(\mathrm{x})$ of configuration x, illustrated on the left in Fig. 2, as

$$\mathcal{LF}(\mathrm{x}) := \left\{ \mathrm{p} \in B\left(\mathrm{x}, \tfrac{R}{2}\right) \,\middle|\, \|\mathrm{p} - \mathrm{x}\| \leq \|\mathrm{p} - \mathrm{s}\|,\ \forall \mathrm{s} \in \mathcal{S}(\mathrm{x}) \right\}. \quad (7)$$

Note that $\mathcal{LF}(\mathrm{x})$ is a closed convex set whose boundary is defined by the maximum-margin separating hyperplanes[5] between x and obstacles [35], [36]. Hence, it is straight forward to observe that:

Proposition 1 *$\mathcal{LF}(\mathrm{x}) \subseteq \mathcal{F}$ for any $\mathrm{x} \in \mathcal{F}$.*

In robotics, a safe neighborhood of a configuration $\mathrm{x} \in \mathcal{F}$ is generally defined as the largest ball centered at x in the free space $\mathcal{F}$, i.e., $B(\mathrm{x}, d(\mathrm{x}, \partial\mathcal{F}))$, which is a very restrictive usage of critical collision sensing since $d(\mathrm{x}, \partial\mathcal{F}) = \min_{\mathrm{s} \in \mathcal{S}(\mathrm{x})} \|\mathrm{s} - \mathrm{x}\|$ for large enough R. Simplicity notwithstanding, such a primitive symmetric safe zone, $B(\mathrm{x}, d(\mathrm{x}, \partial\mathcal{F}))$, around x is, unfortunately, not able to capture the local geometry of the free space $\mathcal{F}$ around x. In contrast, we believe that this new notion of a convex local free space $\mathcal{LF}(\mathrm{x})$ around x may provide a computationally effective local representation of the free space [42], that is useful for incremental motion planning.

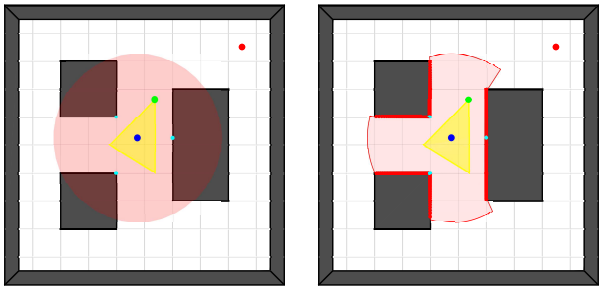


Fig. 2. Local free space $\mathcal{LF}(\mathrm{x})$ (yellow polygon) around a sample configuration $\mathrm{x} \in \mathcal{F}$ (blue point), constructed from critical collisions (cyan points) around x: (left) limited range critical collision "sensing" (6) and (right) limited range radial collision "sensing" (9). To reach a destination configuration $\mathrm{y} \in \mathcal{F}$ (red point), sensory steering generates a step toward the closest point $\Pi_{\mathcal{LF}(\mathrm{x})}(\mathrm{y})$ (green point) of the local free space $\mathcal{LF}(\mathrm{x})$ to the goal y.

2) Local Free Space from Radial Collision Sensing: We now generalize to higher dimensions a version of critical collision sensing that is more practical in low dimensional settings where radial range scanning is literally available (e.g., 2D LIDAR range scanner and 3D depth sensors). Let $\rho : \mathbb{R}^n \times \mathbb{S}^{n-1} \to [0, R]$ be a limited-range radial distance-to-collision function that returns the distance of $\mathrm{x} \in \mathbb{R}^n$ to the boundary $\partial\mathcal{F}$ of the free space $\mathcal{F}$ along direction $\mathrm{v} \in \mathbb{S}^{n-1}$, defined as

$$\rho(\mathrm{x}, \mathrm{v}) := \max \left\{ \alpha \in [0, R] \,\middle|\, [\mathrm{x}, \mathrm{x} + \alpha\mathrm{v}] \subseteq \mathcal{F} \right\}, \quad (8)$$

if $\mathrm{x} \in \mathcal{F}$, and zero otherwise. Recall from (6) that the critical collision point on a convex obstacle is defined to be its closest point to a configuration. Accordingly, using the strict local minima of ρ, we identify critical collision points around x as

$$\widehat{\mathcal{S}}(\mathrm{x}) := \left\{ \mathrm{x} + \rho(\mathrm{x}, \mathrm{v})\, \mathrm{v} \,\middle|\, \mathrm{v} \in \mathcal{M}\rho(\mathrm{x}) \right\}, \quad (9)$$

where the set of the strict local minima of ρ at x is given by

$$\mathcal{M}\rho(\mathrm{x}) := \left\{ \mathrm{v} \in \mathbb{S}^n \,\middle|\, \exists \varepsilon > 0 \text{ s.t. } \rho(\mathrm{x}, \mathrm{v}) < \rho\left(\mathrm{x}, \frac{\mathrm{v} + \alpha\mathrm{u}}{\|\mathrm{v} + \alpha\mathrm{u}\|}\right) \right.$$
$$\left. \forall \alpha \in (0, \varepsilon),\, \mathrm{u} \in \mathbb{S}^{n-1} \right\}. \quad (10)$$

In practice, one can use a regular grid discretization of $\mathbb{S}^{n-1}$ [43] and exhaustively search for the strict local minima of ρ.

Therefore, using critical collision points in $\widehat{\mathcal{S}}(\mathrm{x})$ and the associated Voronoi decomposition of the sensory footprint, we define the local free space $\widehat{\mathcal{LF}}(\mathrm{x})$ of $\mathrm{x} \in \mathcal{F}$ as

$$\widehat{\mathcal{LF}}(\mathrm{x}) := \left\{ \mathrm{p} \in \mathcal{P}(\mathrm{x}) \,\middle|\, \|\mathrm{p} - \mathrm{x}\| \leq \|\mathrm{p} - \mathrm{s}\|,\ \forall \mathrm{s} \in \widehat{\mathcal{S}}(\mathrm{x}) \right\}, \quad (11)$$

where the (half-scale) sensory footprint is given by

$$\mathcal{P}(\mathrm{x}) := \left\{ \mathrm{x} + \frac{1}{2}\alpha\rho(\mathrm{x}, \mathrm{v})\, \mathrm{v} \,\middle|\, \mathrm{v} \in \mathbb{S}^{n-1}, \alpha \in [0, 1] \right\}. \quad (12)$$

Proposition 2 *For any $\mathrm{x} \in \mathcal{F}$, the local free space $\widehat{\mathcal{LF}}(\mathrm{x})$ is a closed subset of $\mathcal{F}$, but not necessarily convex.*[6]

Proof. By definition, $\mathcal{P}(\mathrm{x}) \subseteq \mathcal{F}$, and so $\widehat{\mathcal{LF}}(\mathrm{x}) \subseteq \mathcal{F}$. Further, by construction, both $\mathcal{P}(\mathrm{x})$ and $\widehat{\mathcal{LF}}(\mathrm{x})$ are closed. For a counter example for $\widehat{\mathcal{LF}}(\mathrm{x})$ being convex, see Fig. 3. ■

[4]It is a common practice to represent obstacles as a union of convex polytopes because the surface features that define the closest point between convex polytopes persist under small perturbations and so the closest point between convex polytopes can be computed incrementally [40].

[5]The maximum margin separating hyperplane [41] between any two distinct points $\mathrm{a} \neq \mathrm{b} \in \mathbb{R}^n$ can be equivalently written as

$$\left\{ \mathrm{x} \in \mathbb{R}^n \,\middle|\, \|\mathrm{x} - \mathrm{a}\| = \|\mathrm{x} - \mathrm{b}\| \right\} = \left\{ \mathrm{x} \in \mathbb{R}^n \,\middle|\, (\mathrm{a} - \mathrm{b})^{\mathrm{T}} \left(\mathrm{x} - \tfrac{\mathrm{a}+\mathrm{b}}{2}\right) = 0 \right\}.$$

[6]One can define a convex version of the local free space in (11) as

$$\widehat{\mathcal{LF}}(\mathrm{x}) := \left\{ \mathrm{p} \in B\left(\mathrm{x}, \min\left(\tfrac{R}{2}, d(\mathrm{x}, \partial\mathcal{F})\right)\right) \,\middle|\, \|\mathrm{p} - \mathrm{x}\| \leq \|\mathrm{p} - \mathrm{s}\|,\ \forall \mathrm{s} \in \widehat{\mathcal{S}}(\mathrm{x}) \right\},$$

but the restriction to $B(\mathrm{x}, d(\mathrm{x}, \partial\mathcal{F}))$ might be very conservative near the free space boundary.

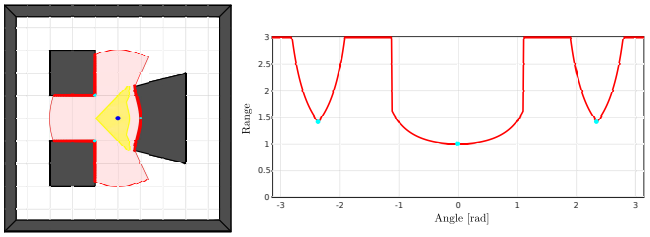


Fig. 3. A nonconvex local free space (yellow region) constructed using the strict local minima (cyan point) and the (half-scale) sensory footprint (red region) of the range map ρ.

Note that to ensure the convexity of $\widehat{\mathcal{LF}}(\mathrm{x})$, instead of the strict local minima of ρ, as discussed in Section III-A.1, one can construct the local free space using convex surface decomposition of the range data [44]. Besides, if the configuration space obstacles can be represented as a finite union of convex sets, then the convexity issue of $\widehat{\mathcal{LF}}(\mathrm{x})$ becomes significantly less severe. In the following, we shall introduce an alternative approach to get around this issue.

B. Sensory Steering Function

Inspired by the "move-to-projected-goal" paradigm introduced in [35], [36], we design our sensory steering function $\texttt{SensorySteer} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ that returns a steering step at a configuration $\mathrm{x} \in \mathcal{F}$ toward a desired configuration $\mathrm{y} \in \mathcal{F}$ through the "projected-goal" $\Pi_{\mathcal{LF}(\mathrm{x})}(\mathrm{y})$ as follows:

$$\texttt{SensorySteer}(\mathrm{x},\mathrm{y}) := \Pi_{B(\mathrm{x},\epsilon)} \circ \Pi_{\mathcal{LF}(\mathrm{x})}(\mathrm{y}), \qquad (13)$$

where $\epsilon > 0$ is a fixed step size, and $\Pi_{\mathcal{LF}(\mathrm{x})}$ denotes the metric projection onto the closed convex local free space $\mathcal{LF}(\mathrm{x})$ defined in (7). Note that metric projection onto a closed convex set is piecewise continuously differentiable [45], and, since $\mathrm{x} \in \mathcal{LF}(\mathrm{x}) \cap B(\mathrm{x},\epsilon)$, by definition, yields a greedy sensory steering function in the sense that

$$\|\mathrm{x}-\mathrm{y}\| \geq \|\texttt{SensorySteer}(\mathrm{x},\mathrm{y})-\mathrm{y}\|. \qquad (14)$$

Our construction of sensory steering in (13) is strongly based on metric projection onto convex sets, which can be efficiently computed using a standard off-the-shelf convex optimization solver [41]. When the local free space $\mathcal{LF}(\mathrm{x})$ is a closed convex polytope, then the projected goal $\Pi_{\mathcal{LF}(\mathrm{x})}(\mathrm{y})$ can be efficiently computed in polynomial time by solving a linearly constrained least squares problem using active set methods [34], briefly presented in Section IV-B. Hence, to ensure polynomial time computational complexity, we find it convenient to redefine our sensory steering function as

$$\overline{\texttt{SensorySteer}}(\mathrm{x},\mathrm{y}) := \Pi_{B\left(\mathrm{x},\min\left(\epsilon,\frac{R}{2}\right)\right)} \circ \Pi_{\mathcal{V}(\mathrm{x})}(\mathrm{y}), \qquad (15a)$$

$$= \mathrm{x} + \min\left(\epsilon, \tfrac{R}{2}, \left\|\Pi_{\mathcal{V}(\mathrm{x})}(\mathrm{y})-\mathrm{x}\right\|\right) \frac{\Pi_{\mathcal{V}(\mathrm{x})}(\mathrm{y})-\mathrm{x}}{\left\|\Pi_{\mathcal{V}(\mathrm{x})}(\mathrm{y})-\mathrm{x}\right\|}, \qquad (15b)$$

where $R > 0$ is the fixed sensing range, $\mathcal{V}(\mathrm{x})$ is the Voronoi cell of x, associated with the critical collision points $\mathcal{S}(\mathrm{x})$ in (6), defined as

$$\mathcal{V}(\mathrm{x}) := \left\{ \mathrm{p} \in \mathbb{R}^n \,\middle|\, \|\mathrm{p}-\mathrm{x}\| \leq \|\mathrm{p}-\mathrm{s}\|, \forall \mathrm{s} \in \mathcal{S}(\mathrm{x}) \right\}. \qquad (16)$$

Here, metric projection onto an Euclidean ball $B(\mathrm{x},r)$ of radius $r \geq 0$ can be analytically computed as

$$\Pi_{B(\mathrm{x},r)}(\mathrm{y}) = \mathrm{x} + \min(r, \|\mathrm{y}-\mathrm{x}\|) \frac{\mathrm{y}-\mathrm{x}}{\|\mathrm{y}-\mathrm{x}\|}, \qquad (17)$$

whereas, since $\mathcal{V}(\mathrm{x})$ is a convex polytope, as aforementioned, the metric projection $\Pi_{\mathcal{V}(\mathrm{x})}$ can be computed in polynomial time, with the number of hyperplane constraints. Also observe that although $\mathcal{LF}(\mathrm{x}) = B\left(\mathrm{x},\frac{R}{2}\right) \cap \mathcal{V}(\mathrm{x})$, unfortunately, $\Pi_{\mathcal{LF}(\mathrm{x})} \neq \Pi_{B\left(\mathrm{x},\frac{R}{2}\right)} \circ \Pi_{\mathcal{V}(\mathrm{x})}$. Hence, the definitions of sensory steering in (13) and (15) are slightly different.

Following the lines of (15), we also define the sensory steering for radial collision sensing of Section III-A.2 as

$$\widehat{\texttt{SensorySteer}}(\mathrm{x},\mathrm{y}) := \min\left(\epsilon, \rho(\mathrm{x},\mathrm{v}), \left\|\Pi_{\widehat{\mathcal{V}}(\mathrm{x})}(\mathrm{y})-\mathrm{x}\right\|\right)\mathrm{v}, \qquad (18)$$

where $\mathrm{v} = \frac{\Pi_{\widehat{\mathcal{V}}(\mathrm{x})}(\mathrm{y})-\mathrm{x}}{\left\|\Pi_{\widehat{\mathcal{V}}(\mathrm{x})}(\mathrm{y})-\mathrm{x}\right\|}$, and ρ is the radial distance-to-collision function in (8), and the Voronoi cell $\widehat{\mathcal{V}}(\mathrm{x})$ of x, associated with the strict local minima $\widehat{\mathcal{S}}(\mathrm{x})$ in (9) of the range map ρ, is defined as

$$\widehat{\mathcal{V}}(\mathrm{x}) := \left\{ \mathrm{p} \in \mathbb{R}^n \,\middle|\, \|\mathrm{p}-\mathrm{x}\| \leq \|\mathrm{p}-\mathrm{s}\|, \forall \mathrm{s} \in \widehat{\mathcal{S}}(\mathrm{x}) \right\}. \qquad (19)$$

We summarize important properties of $\texttt{SensorySteer}$ as:

Theorem 1 *The sensory steering functions in (13),(15) and (18) are all greedy (14) admissible (2) deterministic safe (4) local planners.*

Proof. The greedy and deterministic construction of $\texttt{SensorySteer}$ is due to metric projection onto convex sets; and its admissibility property (see Definition 1) follows from the fact that $B\left(\mathrm{x},\frac{r}{2}\right) \subseteq \mathcal{LF}(\mathrm{x})$ and $B\left(\mathrm{x},\frac{r}{2}\right) \subseteq \widehat{\mathcal{LF}}(\mathrm{x})$ for all $\mathrm{x} \in \mathcal{F}$, where $r = \min(2\epsilon, d(\mathrm{x},\partial\mathcal{F}))$. Moreover, for any $\mathrm{x},\mathrm{y} \in \mathcal{F}$, $\texttt{SensorySteer}$ always generate a safe steering step, i.e., $\texttt{CollisionFree}(\texttt{SensorySteer}(\mathrm{x},\mathrm{y}))$ is true, since both $\mathcal{LF}(\mathrm{x})$ and $\widehat{\mathcal{LF}}(\mathrm{x})$ are subsets of $\mathcal{F}$. ■

Therefore, it is redundant to check the collision safety of sensory steering while constructing PRMs and RRTs in Algorithm 1 and Algorithm 2, respectively, since for any $\mathrm{x},\mathrm{y} \in \mathcal{F}$, $\texttt{CollisionFree}(\texttt{SensorySteer}(\mathrm{x},\mathrm{y}))$ is always true. Hence, adaptive step size selection in (13), (15), and (18) for safe steering intuitively suggests the improvement of sensory steering over the straight-line steering,

$$\texttt{StraightSteer}(\mathrm{x},\mathrm{y}) = \Pi_{B(\mathrm{x},\epsilon)}(\mathrm{y}). \qquad (20)$$

IV. Implementation Details

A. Complexity

We now briefly discuss the complexity of sensory steering in comparison to the computational cost of distance-based collision detection. If the configuration space obstacles are explicitly represented as a finite union of convex sets, $\{Q_1, Q_2, \ldots, Q_m\}$, then the collision detection of a line segment $[\mathrm{x},\mathrm{y}] \subseteq \mathbb{R}^n$ can be performed using

$$\texttt{CollisionFree}([\mathrm{x},\mathrm{y}]) \iff d([\mathrm{x},\mathrm{y}],Q_i) > 0 \quad \forall i, \qquad (21)$$

whose computational cost is given by

$$\mathrm{O}\left(\texttt{CollisionFree}([\mathrm{x},\mathrm{y}])\right) = \sum_{i=1}^{m} \mathrm{O}\left(d([\mathrm{x},\mathrm{y}], Q_i)\right), \quad (22)$$

where $\mathrm{O}\left(d([\mathrm{x},\mathrm{y}], Q_i)\right)$ is the cost of computing the distance between line segment $[\mathrm{x},\mathrm{y}]$ and obstacle Q_i.

Likewise, for any $\mathrm{x},\mathrm{y} \in \mathcal{F}$, the computational complexity of our sensory steering function in (13) can be determined as

$$\mathrm{O}\big(\texttt{SensorySteer}\,(\mathrm{x},\mathrm{y})\big) = \mathrm{O}\big(\mathcal{LF}(\mathrm{x})\big) + \mathrm{O}\big(\Pi_{\mathcal{LF}(\mathrm{x})}(\mathrm{y})\big), \quad (23)$$

$$= \mathrm{O}\big(\Pi_{\mathcal{LF}(\mathrm{x})}(\mathrm{y})\big) + \sum_{i=1}^{m} \mathrm{O}\big(\Pi_{Q_i}(\mathrm{x})\big). \quad (24)$$

$$= \mathrm{O}\big(d(\mathrm{y}, \mathcal{LF}(\mathrm{x}))\big) + \sum_{i=1}^{m} \mathrm{O}\big(d(\mathrm{x}, Q_i)\big), \quad (25)$$

where the latter follows from the assumption that the costs of computing the distance of a point $\mathrm{x} \in \mathbb{R}^n$ and its closest point to a closed convex set $A \subseteq \mathbb{R}^n$ are the same, i.e., $\mathrm{O}\big(d(\mathrm{x}, A)\big) = \mathrm{O}\big(\Pi_A(\mathrm{x})\big)$, because $d(\mathrm{x}, A) = \|\mathrm{x} - \Pi_A(\mathrm{x})\|$.

Therefore, since $\mathrm{O}(d(\mathrm{x}, Q_i)) \leq \mathrm{O}\big(d([\mathrm{x},\mathrm{y}], Q_i)\big)$, one can conclude from (22) and (25) that if the explicit representation of configuration space is available as a union of convex components, then the cost of sensory steering $\mathrm{O}\big(\texttt{SensorySteer}(\mathrm{x},\mathrm{y})\big)$ is generally significantly less than the cost of collision detection of a line segment $\mathrm{O}\big(\texttt{CollisionFree}([\mathrm{x},\mathrm{y}])\big)$.

In case that the configuration space is not available explicitly, sensory steering can be preformed using radial collision sensing of Section III-A.2. If the range map ρ in (8) is assumed to provide a (uniform) resolution of M measurements, then the cost of sensory steering becomes a constant multiple of the cost of distance-based collision detection,

$$\mathrm{O}\left(\texttt{SensorySteer}\right) = M \cdot \mathrm{O}\left(\texttt{CollisionFree}\right). \quad (26)$$

B. Active Set Methods for Convex Quadratic Optimization

In our implementations, we recast metric projection onto convex polytopes and distance between them as convex quadratic optimization problems, and solve them iteratively using the active set method, summarized below.

Consider a convex quadratic optimization problem with equality and inequality constraints (QP-IE):

$$\begin{aligned} \min_{\mathrm{x}\in\mathbb{R}^n} \quad & f(\mathrm{x}) = \frac{1}{2}\mathrm{x}^{\mathrm{T}}\mathbf{Q}\,\mathrm{x} + \mathrm{x}^{\mathrm{T}}\mathrm{c} \\ \text{subject to} \quad & \mathrm{a}_i^{\mathrm{T}}\mathrm{x} = b_i, \;\; i \in \mathcal{E}, \\ & a_j^{\mathrm{T}} x \geq b_j, \;\; j \in \mathcal{I}, \end{aligned} \qquad \text{(QP-IE)}$$

where $\mathbf{Q} \in \mathbb{R}^{n\times n}$ is a positive definite matrix, $\mathcal{E}$ and $\mathcal{I}$ are sets of indices for equality and inequality constraints, respectively, and $\mathrm{c}, \mathrm{a}_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$, where $i \in \mathcal{E} \cup \mathcal{I}$. Also, let $\mathcal{X} := \left\{\mathrm{x} \in \mathbb{R}^n \middle| \mathrm{a}_i{}^{\mathrm{T}}\mathrm{x} = b_i \;\forall i \in \mathcal{E},\; \mathrm{a}_j{}^{\mathrm{T}}\mathrm{x} \geq b_j \;\forall j \in \mathcal{I}\right\}$ denote the set of feasible solutions of (QP-IE), and let $\mathcal{A}\,(\mathrm{x}) := \mathcal{E} \cup \left\{j \in \mathcal{I} \middle| \mathrm{a}_i{}^{\mathrm{T}}\mathrm{x} = b_j\right\}$ be the index set of active constraints at a feasible solution $\mathrm{x} \in \mathcal{X}$.

Among many alternative solvers [34], [41], active set methods offers an iterative solution for the convex quadratic optimization problem (QP-IE), whose iterations, denoted by $\mathrm{x}_{k+1} = \texttt{AS}(\mathrm{x}_k)$, satisfy for any feasible solution $\mathrm{x}_k \in \mathcal{X}$ the properties:

(i) (Feasible Iterations) $\texttt{AS}\,(\mathrm{x}_k) \in X$,
(ii) (Monotonic Decrease) $f\,(\mathrm{x}_k) \geq f\,(\texttt{AS}\,(\mathrm{x}_k))$,
(iii) (Finite-Step Global Convergence) $\texttt{AS}\,(\mathrm{x}_k)$ converges in polynomial steps to the global solution of (QP-IE).

More precisely, to find the global solution of (QP-IE), the active set method starts with a feasible solution $\mathrm{x}_0 \in \mathcal{X}$ and, at each iteration $k \in \mathbb{N}$, it solves an associated convex quadratic optimization problem with equality constraint to find an update step, $\mathrm{p}_k \in \mathbb{R}^n$:

$$\begin{aligned} \min_{\mathrm{p}_k\in\mathbb{R}^n} \quad & \frac{1}{2}\mathrm{p}_k{}^{\mathrm{T}}\mathbf{Q}\,\mathrm{p}_k + \mathrm{p}_k^{\mathrm{T}}\mathrm{g}_k \\ \text{subject to} \quad & \mathrm{a}_i^{\mathrm{T}}\mathrm{p}_k = 0, \;\; i \in \mathcal{W}_k \end{aligned} \qquad \text{(QP-EQ)}$$

where $\mathrm{g}_k = \mathbf{Q}\mathrm{x}_k + \mathrm{c}$ and $\mathcal{W}_k \subseteq \mathcal{A}\,(\mathrm{x}_k)$ is a subset of the indices of the active constraints at x_k with linearly independent constraint gradients, a_i's, and is referred to as the *working set*. The solution to (QP-EQ), denoted by $(\mathrm{p}_k, \lambda_k) = \texttt{SolveQPEQ}\,(\mathrm{x}_k, \mathcal{W}_k)$, can be found by solving

$$\begin{bmatrix} \mathbf{Q} & \mathbf{A}^{\mathrm{T}} \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \begin{bmatrix} -\mathrm{p}_k \\ \lambda_k \end{bmatrix} = \begin{bmatrix} \mathrm{g}_k \\ 0 \end{bmatrix}, \quad (27)$$

where $\mathbf{A} = [\mathrm{a}_i{}^{\mathrm{T}}]_{i\in\mathcal{W}(x_k)}$ is the Jacobian of working set constraints, and λ_k denotes the vector of Lagrange multipliers for (QP-EQ) and is used to check the Karush-Kuhn-Tucker (KKT) optimality condition of x_k for (QP-IE).

In summary, the active set method repeatedly uses the solution of (QP-EQ) to generate a new estimated solution x_{k+1} for (QP-IE), and terminates at the global solution of (QP-EQ), as shown in Algorithm 3. For more details, refer to [34][Chapter 16].

[7]The working set $\mathcal{W}_k$ should always contain the indices of linearly independent constraint gradients and so should be updated accordingly.

Algorithm 3: The Active Set Algorithm [34][Chapter 16]

Input: $\mathrm{x}_0 \in \mathcal{X}$ – Initial Feasible Solution
$\mathcal{W}_0 \subset \mathcal{A}\,(\mathrm{x}_0)$ – Initial Working Set
Output: $\mathrm{x}^* \in \mathcal{X}$ – The global solution of (QP-IE)

for $k = 0, 1, \ldots$ **do**
  $(\mathrm{p}_k, \lambda) \leftarrow \texttt{SolveQPEQ}\,(\mathrm{x}_k, \mathcal{W}_k)$;
  if $\mathrm{p}_k = 0$ **then**
    if $\lambda_i \geq 0 \;\; \forall i \in \mathcal{W}_k \cap \mathcal{I}$ **then**
      $\mathrm{x}^* \leftarrow \mathrm{x}_k$; **return** x^*;
    else
      $j \leftarrow \arg\min_{j\in\mathcal{W}_k\cap\mathcal{I}} \lambda_j$;
      $\mathrm{x}_{k+1} \leftarrow \mathrm{x}_k$; $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{j\}$;
  else
    $\alpha_k \leftarrow \min\left(1, \min_{i\in\mathcal{I}\setminus\mathcal{W}_k,\; \mathrm{a}_i^{\mathrm{T}}\mathrm{p}_k<0} \frac{b_i - \mathrm{a}_i^{\mathrm{T}}\mathrm{x}_k}{\mathrm{a}_i^{\mathrm{T}}\mathrm{p}_k}\right)$;
    $\mathrm{x}_{k+1} \leftarrow \mathrm{x}_k + \alpha_k \mathrm{p}_k$;
    if $\exists i \in \mathcal{A}\,(\mathrm{x}_{k+1}) \setminus \mathcal{W}_k$ **then**[7] $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \cup \{i\}$;
    else $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k$;

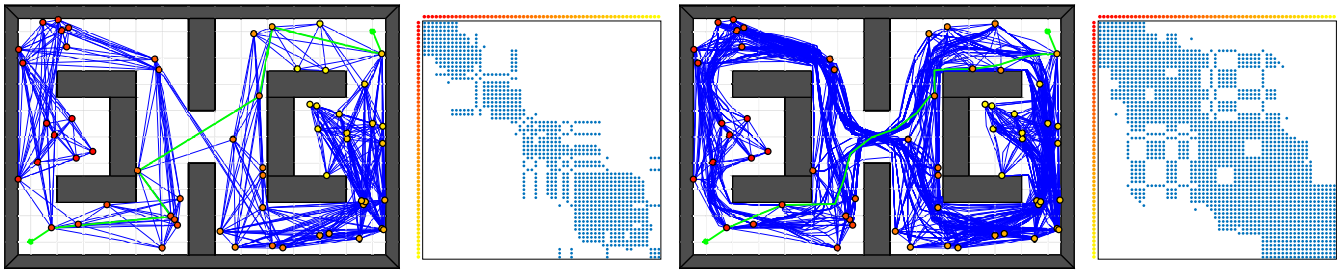


Fig. 4. Probabilistic roadmaps (PRMs) and their adjacency matrices obtained using (left) the straight-line planner, (right) our sensory steering algorithm. The green path shows a path found via a PRM that joins a start and a goal configurations (green points). Here, we set the number of samples to $N = 60$, the number of steering steps to $K = 20$, and the maximum step size to $\epsilon = 1$ unit.

To conclude this section, we emphasize a virtue of the active set method beyond its polynomial time complexity. We believe its feasible iterations and guaranteed monotonic decrease make it a compelling option for general incremental anytime computations of distance between convex bodies, metric projection onto convex sets, and, of course, our sensory steering function. This anytime nature affords opportunistic interruption of its computation while relying on the last iterated feasible solution as an estimate of the global optimal solution. In the context of dynamically evolving motion planning or dynamic settings, these interruptions can be event based and the results of the previous computation can improve the initiation of its successor.

V. Numerical Simulations

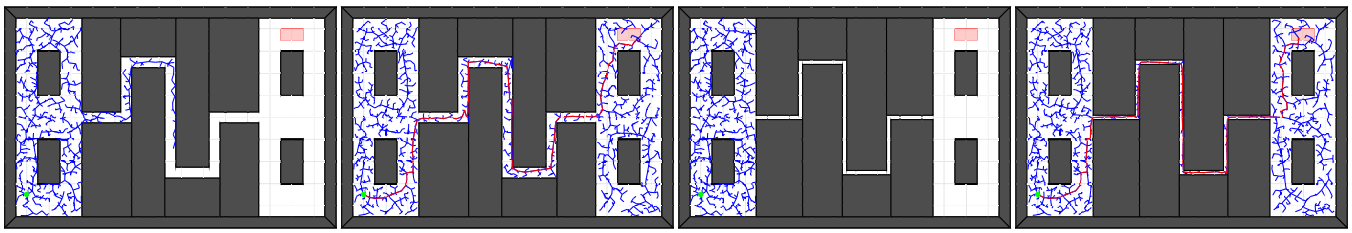
In this section, we provide simulation results demonstrating the effectiveness of the proposed sensory steering over the standard straight-line planner for increased connectivity of random motion planing graphs.

In Fig. 4, for a 2D configuration space with overlapping convex obstacles, we compare the probabilistic roadmaps (PRMs) constructed using the straight-line planner and our sensory steering algorithm. Here, we uniformly generate $N = 60$ sample configurations from the free space using rejection sampling, and join two sample configurations using at most $K = 20$ steering steps of maximum step size $\epsilon = 1$ unit. As seen in Fig. 4, sensory steering generally generates more complex but effective paths to join sample configurations than the straight-line planner does, which explains the significant improvement in the connectivity of resulting PRMs, as clearly observed from the associated adjacency matrices. It is worth mention that in our simulation studies we observe that when the configuration space obstacles consist of nonoverlapping convex sets, then the sensory steering does significantly better in capturing the local connectivity of the configuration space.

To compare their path finding performance around narrow passages, in Fig. 5 we present the rapidly-exploring random trees (RRTs) constructed using the straight-line planner and our sensory steering function in 2D configuration spaces with maze-like narrow passages of gap size 0.5 and 0.2 units. Here, we set the number of RRT iterations to $N = 1500$ and the maximum step size to $\epsilon = 0.3$ units. Although, the straight-line planner is able to locate the entrance of the narrow passage for the gap size 0.5 units and make some progress along it, it is not able to construct an RRT that connects the start configuration (green point) to the goal region (red polygon) in $N = 1500$ iterations, while our sensory steering yields an RRT that expands to the entire configuration space and finds a path between the start configuration and the goal region. In our simulations, we observe that nearly half of the attempts to expand an RRT using the straight-line planner fails; for example, the RRTs in Fig. 5 (a) and Fig. 5(c) have 734 and 714 vertices, respectively, after $N = 1500$ iterations. Whereas, our anecdotal experience with the sensory steering method gives the impression that it successfully grows an RRT at almost all attempts (Theorem 1), and so the RRTs in Fig. 5 (b,d) both have 1500 vertices, which is due to the fact that, by construction, sensory steering uses adaptive step size in response to the "sensed" local geometry of configuration spaces.

VI. Conclusions

In this paper, we introduce a new deterministic greedy admissible sensory steering algorithm (Theorem 1) for sampling-based motion planning algorithms that significantly increases the connectivity of random motion planning graphs,



(a) (b) (c) (d)

Fig. 5. Rapidly-exploring random trees (RRTs) constructed using (a,c) the straight-line planner and (b,d) our sensory steering algorithm. The red path, if found, is the shortest path of the constructed RRT from the green starting point to the red goal region. Here, we set the number of RRT iterations to $N = 1500$ and the maximum steering step size to $\epsilon = 0.3$ units, and the sizes of narrow gaps in (a,b) and (c,d) are 0.5 and 0.2 units, respectively. Please see the accompanying video submission for an animated demonstration.

especially around difficult regions of configuration spaces such as narrow passages. The construction of our sensory steering algorithm is based on the identification of a local (convex) collision free zone around a sample configuration that reflects the surrounding local geometric structure of a configuration space. Accordingly, our sensory steering algorithm generates a safe steering motion toward the closest point of the local free space of a sample configuration to any given destination configuration, which yields an adaptive step size selection. For an incremental iterative anytime computation of our sensory steering algorithm, we suggest using the active set method of convex quadratic optimization. The effectiveness of the proposed sensory steering is suggested using nontrivial numerical simulations.

Work now in progress targets computationally efficient adaptation of sensory steering to complex configuration spaces, e.g., for manipulator motion planning, using simultaneous optimization of control and sensing. We are also exploring an alternative use of active set methods for incremental distance computation in robot motion planning.

REFERENCES

[1] E. Guizzo, "So, where are my robot servants?" *IEEE Spectrum*, vol. 51, no. 6, pp. 74–79, 2014.

[2] R. D'Andrea, "Guest editorial can drones deliver?" *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 647–648, 2014.

[3] E. Guizzo, "Three engineers, hundreds of robots, one warehouse," *IEEE Spectrum*, vol. 45, no. 7, pp. 26–34, 2008.

[4] P. E. Ross, "Robot, you can drive my car," *IEEE Spectrum*, vol. 51, no. 6, pp. 60–90, 2014.

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

[6] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University, Tech. Rep., 1998.

[7] D. Hsu, L. E. Kavraki, J.-C. Latombe, R. Motwani, and S. Sorkin, "On finding narrow passages with probabilistic roadmap planners," in *The Int. Workshop on the Algorithmic Foundations of Robotics*, 1998.

[8] S. R. Lindemann and S. M. LaValle, "Current issues in sampling-based motion planning," in *Int. Symp. Robot. Res.*, 2005, pp. 36–54.

[9] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, 2001.

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

[11] C. Holleman and L. E. Kavraki, "A framework for using the workspace medial axis in PRM planners," in *IEEE International Conference on Robotics and Automation*, vol. 2, 2000, pp. 1408–1413.

[12] J.-M. Lien, S. L. Thomas, and N. M. Amato, "A general framework for sampling on the medial axis of the free space," in *IEEE International Conference on Robotics and Automation*, vol. 3, 2003, pp. 4439–4444.

[13] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, "OBPRM: An obstacle-based PRM for 3D workspaces," in *Workshop on the Algorithmic Foundations of Robotics*, 1998, pp. 155–168.

[14] J. P. van den Berg and M. H. Overmars, "Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners," *Int. J. Robot. Res.*, vol. 24, no. 12, pp. 1055–1071, 2005.

[15] H. Kurniawati and D. Hsu, "Workspace importance sampling for probabilistic roadmap planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, 2004, pp. 1618–1623.

[16] M. Foskey, M. Garber, M. C. Lin, and D. Manocha, "A Voronoi-based hybrid motion planner," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2001, pp. 55–60.

[17] M. Nowakiewicz, "MST-based method for 6DOF rigid body motion planning in narrow passages," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 5380–5385.

[18] D. Hsu, T. Jiang, J. Reif, and Z. Sun, "The bridge test for sampling narrow passages with probabilistic roadmap planners," in *IEEE Int. Conf. Robot. Autom.*, vol. 3, 2003, pp. 4420–4426.

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

[20] S. Rodriguez, S. Thomas, R. Pearce, and N. M. Amato, "Resampl: A region-sensitive adaptive motion planner," in *Algorithmic Foundation of Robotics VII*. Springer, 2008, pp. 285–300.

[21] D. Aarno, D. Kragic, and H. I. Christensen, "Artificial potential biased probabilistic roadmap method," in *IEEE International Conference on Robotics and Automation*, vol. 1, 2004, pp. 461–466.

[22] J. Denny, R. Sandström, N. Julian, and N. M. Amato, "A region-based strategy for collaborative roadmap construction," in *Algorithmic Foundations of Robotics XI*, 2015, pp. 125–141.

[23] J. Denny and N. M. Amato, "Toggle PRM: Simultaneous mapping of C-free and C-obstacle - a study in 2D -," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 2632–2639.

[24] S. W. H. Wong and M. Jenkin, "Exploiting collision information in probabilistic roadmap planning," in *IEEE International Conference on Mechatronics*, 2009, pp. 1–5.

[25] D. Hsu, G. Sanchez-Ante, and Z. Sun, "Hybrid PRM sampling with a cost-sensitive adaptive strategy," in *IEEE International Conference on Robotics and Automation*, 2005, pp. 3874–3880.

[26] K. E. Bekris, B. Y. Chen, A. M. Ladd, E. Plaku, and L. E. Kavraki, "Multiple query probabilistic roadmap planning using single query planning primitives," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2003, pp. 656–661.

[27] K. Shi, J. Denny, and N. M. Amato, "Spark PRM: Using RRTs within PRMs to efficiently explore narrow passages," in *IEEE International Conference on Robotics and Automation*, 2014, pp. 4659–4666.

[28] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, "Choosing good distance metrics and local planners for probabilistic roadmap methods," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 4, pp. 442–447, 2000.

[29] P. Isto, "Constructing probabilistic roadmaps with powerful local planning and path optimization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2002, pp. 2323–2328.

[30] J. Denny and N. M. Amato, "The toggle local planner for sampling-based motion planning," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 1779–1786.

[31] J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *The International Journal of Robotics Research*, vol. 10, no. 6, pp. 628–649, 1991.

[32] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robot. Res.*, vol. 5, no. 1, pp. 90–98, 1986.

[33] D. Koditschek, "Exact robot navigation by means of potential functions: Some topological considerations," in *IEEE International Conference on Robotics and Automation*, vol. 4, 1987, pp. 1–6.

[34] S. Wright and J. Nocedal, *Numerical Optimization*. Springer, 1999.

[35] O. Arslan and D. E. Koditschek, "Exact robot navigation using power diagrams," in *IEEE Int. Conf. Robot. Autom.*, 2016, pp. 1–8.

[36] ——, "Sensor-based reactive navigation in unknown convex sphere worlds," in *The International Workshop on the Algoritmic Foundations of Robotics*, 2016.

[37] J. J. Kuffner, "Effective sampling and distance metrics for 3D rigid body path planning," in *IEEE International Conference on Robotics and Automation*, vol. 4, 2004, pp. 3993–3998.

[38] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[39] L. E. Kavraki, M. N. Kolountzakis, and J. C. Latombe, "Analysis of probabilistic roadmaps for path planning," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, pp. 166–171, 1998.

[40] M. C. Lin and J. F. Canny, "A fast algorithm for incremental distance calculation," in *IEEE Int. Conf. Robot. Autom.*, 1991, pp. 1008–1014.

[41] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[42] O. Arslan, "Clustering-based robot navigation and control," Ph.D. dissertation, University of Pennsylvania, 2016.

[43] A. Yershova and S. M. LaValle, "Deterministic sampling methods for spheres and SO(3)," in *IEEE International Conference on Robotics and Automation*, vol. 4, 2004, pp. 3974–3980.

[44] S. A. Ehmann and M. C. Lin, "Accurate and fast proximity queries between polyhedra using convex surface decomposition," *Computer Graphics Forum*, vol. 20, no. 3, pp. 500–511, 2001.

[45] A. Shapiro, "Sensitivity analysis of nonlinear programs and differentiability properties of metric projections," *SIAM Journal on Control and Optimization*, vol. 26, no. 3, pp. 628–645, 1988.