The International Journal of Robotics Research
42(4-5):150–175

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Asymptotically optimal inspection planning via efficient near-optimal search on sampled roadmaps

Mengyu Fu[1] , Alan Kuntz[2] , Oren Salzman[3] and Ron Alterovitz[1]

Abstract

Inspection planning, the task of planning motions for a robot that enable it to inspect a set of points of interest, has applications in domains such as industrial, field, and medical robotics. Inspection planning can be computationally challenging, as the search space over motion plans grows exponentially with the number of points of interest to inspect. We propose a novel method, Incremental Random Inspection-roadmap Search (IRIS), that computes inspection plans whose length and set of successfully inspected points asymptotically converge to those of an optimal inspection plan. IRIS incrementally densifies a motion-planning roadmap using a sampling-based algorithm and performs efficient near-optimal graph search over the resulting roadmap as it is generated. We prove the resulting algorithm is asymptotically optimal under very general assumptions about the robot and the environment. We demonstrate IRIS's efficacy on a simulated inspection task with a planar 5 DOF manipulator, on a simulated bridge inspection task with an Unmanned Aerial Vehicle (UAV), and on a medical endoscopic inspection task for a continuum parallel surgical robot in cluttered human anatomy. In all these systems IRIS computes higher-quality inspection plans orders of magnitudes faster than a prior state-of-the-art method.

Keywords

Inspection planning, coverage planning, motion planning

1 Introduction

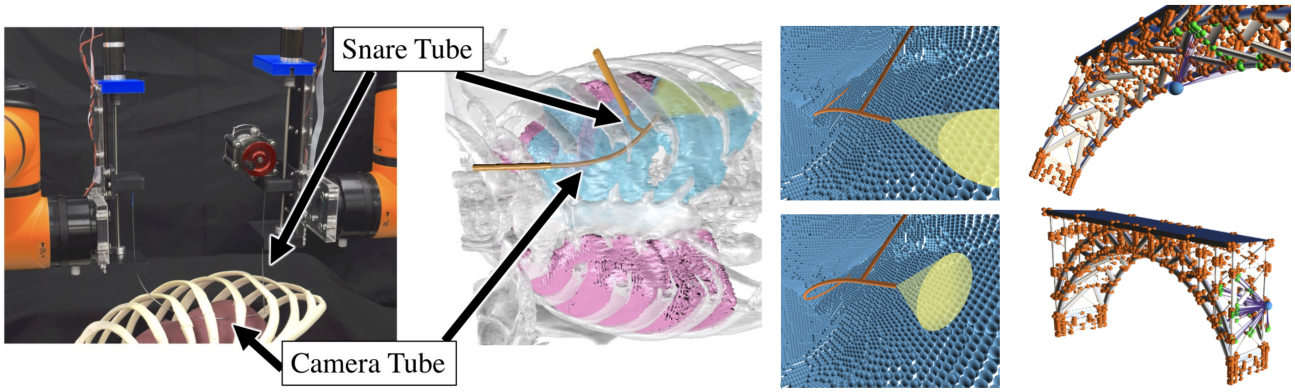
In this work, we investigate the problem of *inspection planning*, or coverage planning (Almadhoun et al. 2016; Galceran and Carreras 2013). Here, we consider the specific setting where we are given a robot equipped with a sensor and a set of points of interest (POI) in the environment to be inspected by the sensor. The problem calls for computing a minimal-length motion plan for the robot that maximizes the number of POI inspected. This problem has a multitude of diverse applications, including surface inspections in industrial production lines (Raffaeli et al. 2013), surveying the ocean floor by autonomous underwater vehicles (Bingham et al. 2010; Gracias et al. 2013; Johnson-Roberson et al. 2010; Tivey et al. 1997), structural inspection of bridges using aerial robots (Bircher et al. 2015, 2016), and medical applications such as inspecting patient anatomy during surgical procedures (Kuntz et al. 2018).

Naïvely computed inspection plans may enable inspection of only a subset of the POI and may require motion plans orders of magnitude longer than an optimal plan, and hence may be undesirable or infeasible due to a robot's battery capacity or time constraints. In medical applications, physicians may want to maximize the number of POI inspected for diagnostic purposes. Additionally, the procedure should be completed as fast as is safely possible to reduce costs and improve patient outcomes, especially if the patient is under anesthesia during the procedure. For example, a robot assisting in the diagnosis of the cause of a pleural effusion (a serious medical condition that can cause the collapse of a patient's lung) will need to visually inspect the surface of the collapsed lung and chest wall inside the body in as short a time as possible (see Fig. 1a). In structural inspecting applications, Unmanned Aerial Vehicles (UAVs), or drones, can be used to efficiently inspect the complex geometry of built structures. High-quality inspection plans could reduce inspection time and reduce costs. Bridge inspection (see Fig. 1b), for example, is critical to ensuring bridge safety since almost 40% of the bridges in the United States exceed their 50-year design life (ASCE 2016). We note that it may not be possible to inspect some POI due to obstacles in the environment and the kinematic constraints of the robot. Our goal is to compute kinematically feasible collision-free inspection plans that maximize the number of POI inspected, and of the motion plans that inspect those POI we compute a shortest plan.

Inspection planning is computationally challenging because the search space is embedded in a high-dimensional *configuration space* $\mathcal{X}$ (the space of all parameters that determine the shape of the robot) (Choset et al. 2005; Latombe 1991; LaValle 2006). Even finding the shortest plan between two points in $\mathcal{X}$ that avoid obstacles (without

[1]Department of Computer Science, University of North Carolina at Chapel Hill, USA
[2]Robotics Center and Kahlert School of Computing, University of Utah, USA
[3]Computer Science Department, Technion - Israel Institute of Technology, Israel

Corresponding author:
Mengyu Fu, Sitterson Hall, 201 S. Columbia Street, Chapel Hill, North Carolina 27599, USA.
Email: mfu@cs.unc.edu



(a) Human anatomy inspection. **(b)** Bridge structure inspection.

Figure 1. Examples of applications of inspection planning. (a) Inspection planning in human anatomy. Left: The Continuum Reconfigurable Incisionless Surgical Parallel (CRISP) robot (Anderson et al. 2017; Mahoney et al. 2016) is composed of needle-diameter tubes assembled into a parallel structure inside the patient's body (in which a tube uses a snare system to grip a tube with a camera affixed to its tip) and then robotically manipulated outside the body, allowing for smaller incisions and faster recovery times compared to traditional endoscopic tools (which have larger diameters). Middle: The CRISP robot in simulation inspecting a collapsed lung, a scenario segmented from a CT scan of a real patient with this condition. The visualization shows the robot (orange), the lungs (pink), and the pleural surface visible (green) and not visible (blue) by the robot's camera sensor in its current configuration. Right: Two example configurations with inspected POI. The CRISP robot (orange) inspects POI (blue) on the organ surface, and visible points are covered by the cone shape (yellow). (b) Inspection planning for bridge structures. A UAV, shown as a blue sphere, inspects points on the surface of a bridge structure. At a configuration, some points are visible (shown in green) while other points are not visible (shown in orange).

reasoning about inspection) is computationally hard.* If we want to compute a minimum-length motion plan that maximizes the number of POI inspected, the complexity of our problem is increased as we have to simultaneously reason about the system's constraints, motion plan length, and POI inspected.

There are multiple approaches to computing inspection plans. Optimization-based methods locally search over the space of all inspection plans (Bircher et al. 2015; Bogaerts et al. 2018). Decoupled approaches first independently select suitable viewpoints and then determine a visiting sequence, i.e., a motion plan for the robot that realizes this sequence (Danner and Kavraki 2000; Englot and Hover 2011). Finally, recent progress in motion planning (Karaman and Frazzoli 2011) has enabled methods to exhaustively search over the space of all motion plans (Bircher et al. 2017; Kafka et al. 2016; Papadopoulos et al. 2013) thus guaranteeing asymptotic optimality, an important feature in many applications, including medical ones. Roughly speaking, asymptotic optimality for inspection planning means these methods produce inspection plans whose length and the number of points inspected will asymptotically converge to those of an optimal inspection plan, given enough planning time.

Of all the aforementioned methods, only algorithms in the latter group provide any formal guarantees on the quality of the solution. This guarantee is achieved by attempting to exhaustively compute the set of Pareto-optimal inspection plans embedded in $\mathcal{X}$ for which full coverage has not been obtained. Namely, for every configuration $x \in \mathcal{X}$, they (asymptotically) compute the set of paths Π_x starting from a given start configuration x_s such that $\forall \pi_1, \pi_2 \in \Pi_x$, either π_1 is shorter than π_2 and π_2 covers POI not covered by π_1 or vice versa. Once Π_x contains a path π_x^* that covers all POI, this path is considered as a candidate solution. In our setting, the set of Pareto-optimal inspection plans is the minimal set of inspection plans such that each plan is either shorter or has better coverage of the POI than any other inspection plan.† Unfortunately, this comes at the price of very long computation times as the size of the search space is exponential in the number of POI.

To this end, we introduce Incremental Random Inspection-roadmap Search (IRIS), a new asymptotically optimal inspection-planning algorithm. IRIS incrementally constructs a sequence of increasingly dense roadmaps—graphs embedded in $\mathcal{X}$ wherein each vertex represents a collision-free configuration and each edge a collision-free transition between configurations—and computes an inspection plan on the roadmaps as they are constructed (see Fig. 2).

Unfortunately, even the problem of computing an optimal inspection plan on a graph (and not in the continuous space) is computationally hard. To this end, our key insight is to compute a *near-optimal* inspection plan on each roadmap. Namely, we compute an inspection plan that is at most $1 + \varepsilon$ the length of an optimal plan while covering at least p-percent of the number of POI (for any $\varepsilon \geq 0$ and $p \in (0, 1]$). This additional flexibility allows us to improve the quality of our inspection plan in an *anytime* manner, i.e., the algorithm can be stopped at any time and return the best inspection plan found up until that point. We achieve this by incrementally densifying the roadmap and then searching over the densified roadmap for a near-optimal inspection plan—a process that

*Computing the shortest motion plan for a point robot moving amidst polyhedral obstacles in 3D is NP-hard, and many variants of the general motion planning problem are PSPACE-hard. For further details, see Halperin et al. (2018).

†More formally, an inspection plan P connecting two configurations $\mathbf{q}, \mathbf{q}' \in \mathcal{X}$ is said to be Pareto optimal in our setting if any other plan connecting $\mathbf{q}$ to $\mathbf{q}'$ is either longer or does not inspect a point visible to P.

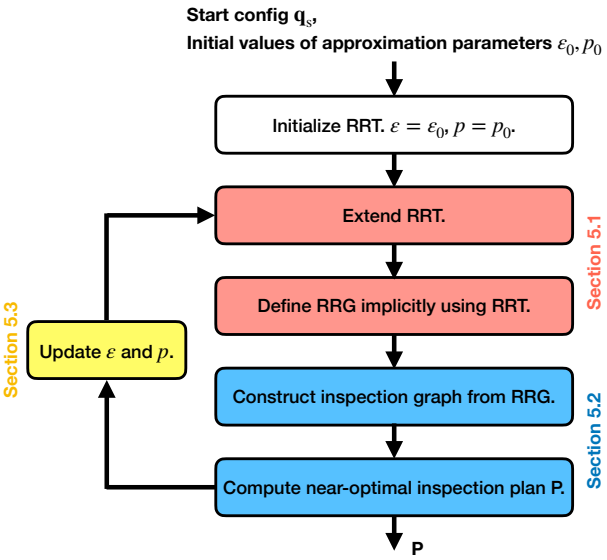


Figure 2. Overview of the IRIS algorithmic framework.

is repeated as time allows. By reducing the approximation factor between iterations, we ensure that our method is asymptotically optimal.

The key contribution of our work is a computationally efficient algorithm to compute provably near-optimal inspection plans on graphs. Coupled with our method for generating this graph, this algorithmic building block enables us to dramatically outperform Rapidly-exploring Random Tree Of Trees (RRTOT) (Bircher et al. 2017)—a state-of-the-art asymptotically optimal inspection planner. Specifically, we demonstrate the efficacy of our approach in simulation for several complex robotic systems (Fig. 1) , including a continuum robot with complex kinematics—the needle-diameter Continuum Reconfigurable Incisionless Surgical Parallel (CRISP) robot (Anderson et al. 2017; Mahoney et al. 2016), working in a medically inspired setting involving the diagnosis of a pleural effusion.

In this paper, we are providing a refined version of our results in Fu et al. (2019), and two important extensions. First, we show results for a simulated bridge inspection task with a UAV, where the inspection target has a complex structure and the underlying roadmap to compute a near-optimal inspection plan on is much larger (e.g. having more nodes and edges). Second, we prove that IRIS is asymptotically optimal under very general system and environment assumptions. These extensions show that IRIS can be used for general complex-structure inspection while providing provable guarantees.

2 Related Work

2.1 Sampling-based motion planning

Motion planning algorithms aim to compute a collision-free motion for a robot to accomplish a task in an environment cluttered with obstacles (Halperin et al. 2018; LaValle 2006; Lynch and Park 2017). A common approach to motion planning is by *sampling-based* algorithms that construct a *roadmap*. Examples include the Probabilistic Roadmaps (PRMs) (Kavraki et al. 1996) (often for solving multiple queries) and the Rapidly-exploring Random Trees (RRTs) (LaValle and Kuffner 2001) for solving single queries. These methods, and many variations thereof, are *probabilistically complete*—namely, the likelihood that they will find a solution, if one exists, approaches certainty as computation time increases.

Recent variations of these methods, such as PRM* and RRT* (Karaman and Frazzoli 2011), improve upon this guarantee by exhibiting *asymptotic optimality*—namely that the quality of the solution obtained, given some cost function, approaches the global optimum as computation increases. Roughly speaking, this is achieved by increasing the (potential) edge set of roadmap vertices considered as its size increases (Karaman and Frazzoli 2011; Solovey et al. 2018). One such algorithm is the Rapidly-exploring Random Graphs (RRGs) (Karaman and Frazzoli 2011) which will be used in our work. RRG combines the exploration strategy of RRT with an updated connection strategy that allows for cycles in the roadmap. It requires solving the two-point boundary value problem (LaValle 2006), which is only efficiently solvable for some robotic systems (including ours).

Guaranteeing asymptotic optimality can come with a heavy computational cost. This inspired work on planners that trade asymptotic optimality guarantees with *asymptotic near optimality* (e.g., Li et al. (2016); Marble and Bekris (2011); Salzman and Halperin (2016)). Asymptotic near optimality states that given an approximation factor $\varepsilon \geq 0$, the solution obtained converges to within a factor of $(1+\varepsilon)$ of the optimal solution with probability one, as the number of samples tends to infinity. Relaxing optimality to near optimality allows a method to improve the practical convergence rate while maintaining desired theoretic guarantees on the quality of the solution.

2.2 Inspection planning

Many inspection-planning algorithms, or coverage planners, decompose the region containing the POI into multiple sub-regions and then solve each sub-region separately (Galceran and Carreras 2013). These methods have limitations, however, such as when occlusions play a significant role in the inspection (Englot and Hover 2012), or when kinematic constraints must be considered (Edelkamp et al. 2017).

Other approaches simultaneously consider all POI. One approach decouples the problem into the coverage sampling problem (CSP) and the multi-goal planning problem (MPP), and solves each independently (Bircher et al. 2015; Danner and Kavraki 2000; Edelkamp et al. 2017; Englot and Hover 2012, 2011). In CSP, a minimal set of viewpoints that provide full inspection coverage is computed. In MPP, a shortest tour that connects all the viewpoints is computed. These correspond to solving the art gallery problem and the traveling salesman problem, respectively. Several of these variants have been shown to be probabilistically complete (Englot and Hover 2012), but none provide guarantees on the quality of the final solution.

The set of viewpoints and the inspection plan itself can also be generated simultaneously. Papadopoulos et al. (2013) propose the Random Inspection Tree Algorithm (RITA). RITA takes into account the differential constraints of the robot and computes both target points for inspection and the trajectory to visit the targets simultaneously. Bircher et al.

(2017) propose Rapidly-exploring Random Tree Of Trees (RRTOT) which constructs a meta–tree structure consisting of multiple RRT* trees. Both methods, which were shown to be asymptotically optimal, iteratively generate a tree, in which the inspection plan is enforced to be a plan from the root to a leaf node. However, each inspection plan does not consider configurations from other branches in the tree which may cause long planning times. This motivates our RRG-based approach.

2.3 Path planning on graphs

Planning a minimal-cost path on a graph is a well-studied problem. When the cost function has an optimal substructure (namely, when subpaths of an optimal path are also optimal), efficient algorithms such as Dijkstra (Dijkstra 1959), A* (Hart et al. 1968), and the many variants thereof can be used. However, in certain settings, including ours, this is not the case. For example, Tsaggouris and Zaroliagis (2004) consider the case where every edge has two attributes (e.g., cost and resource), and the cost function incorporates the attributes in a non-linear fashion.

It is worth mentioning the idea of progressively tightening the approximation factor was also adopted in some anytime A* algorithms based on weighted A* (Pohl 1970), including Anytime Repairing A* (ARA*) by Likhachev et al. (2003) and Restarting Weighted A* (RWA*) by Richter et al. (2010). Anytime Nonparametric A* (ANA*) by van den Berg et al. (2011) furthermore gets rid of the explicit approximation parameter and performs solution improvement adaptively. Nevertheless, as these methods are based on weighted A*, a prerequisite for good performance is a high-quality heuristic, which is not easy to obtain in the case of inspection planning due to the lack of optimal substructure. Furthermore, these methods focus on static graphs (while in our case the graph is incrementally updated) and consider only a single objective (while in our case we have two objectives, inspection and path length). When looking at multiple objectives (though still considering static graphs), recent work by Zhang et al. (2022) extends the approach presented in this paper to suggest an anytime approximate bi-criteria search algorithm.

Inspection planning also bears resemblance to multi-objective path planning. Here, we are given a set of cost functions and are required to find the set of Pareto-optimal paths (Pardalos et al. 2008). Unfortunately, this set may be exponential in the problem size (Ehrgott and Gandibleux 2000). However, it is possible to compute a fully polynomial-time approximation scheme (FPTAS) for many cases (Tsaggouris and Zaroliagis 2009). For additional results on path planning with multiple objectives or when the cost function does not have an optimal substructure, see e.g., (Chen and Nie 2013; Reinhardt and Pisinger 2011; Hernández et al. 2023) and references within.

3 Problem Definition

In this section, we formally define the inspection planning problem. The robot operates in a workspace $\mathcal{W} \subset \mathbb{R}^3$ amidst a set of obstacles $\mathcal{W}_{\text{obs}} \subset \mathcal{W}$. The robot's configuration $\mathbf{q}$ is a d-dimensional vector that uniquely defines the shape of the robot (including, for example, rotation angles and translational extension of all joints). The set of all such configurations is the configuration space $\mathcal{X}$. The geometry of the robot is a configuration-dependent shape $\text{Shape}(\mathbf{q}) \subset \mathcal{W}$, here, $\text{Shape}(\cdot)$ is a mapping from configuration space $\mathcal{X}$ to workspace $\mathcal{W}$, determining the subset of the workspace that the robot occupies for a given configuration. We say that $\mathbf{q} \in \mathcal{X}$ is *in collision* if $\text{Shape}(\mathbf{q}) \cap \mathcal{W}_{\text{obs}} \neq \emptyset$. Let $\mathcal{X}_{\text{obs}} \subseteq \mathcal{X}$ be the obstacle space, such that $\mathcal{X} \setminus \mathcal{X}_{\text{obs}}$ is an open set. Then the *collision free* space is defined as $\mathcal{X}_{\text{free}} = \text{cl}(\mathcal{X} \setminus \mathcal{X}_{\text{obs}})$, where $\text{cl}(\cdot)$ is closure of a set. In this work we define a motion plan for the robot as a path P in $\mathcal{X}$, which is represented as a sequence of n configurations $\{\mathbf{q}_0, \ldots, \mathbf{q}_{n-1}\}$ (vertices) connected by straight-line segments (edges) in $\mathcal{X}$. And we say that P is collision-free if all configurations along P (vertices and edges) are collision-free. We assume that we have a distance function $\ell : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and denote the length of a path P as the sum of the distances between consecutive vertices, i.e., $\ell(P) := \sum_i \ell(\mathbf{q}_i, \mathbf{q}_{i-1})$.

We assume that the robot is equipped with a sensor $\mathcal{S}$ and we are given a set of k points of interest (POI) $\mathcal{I} = \{i_1, \ldots, i_k\}$ in $\mathcal{W}$. We model the sensor as a mapping $\mathcal{S} : \mathcal{X} \to 2^{\mathcal{I}}$, where $2^{\mathcal{I}}$ is the power set of $\mathcal{I}$ and S denotes the subset of $\mathcal{I}$ that can be inspected from each configuration. By a slight abuse of notation, given a path P we set $\mathcal{S}(P) := \bigcup_{i=0}^{n-1} \mathcal{S}(\mathbf{q}_i)$ and note that in our model, we only inspect $\mathcal{I}$ along the vertices of a path.

Definition 1. Inspection coverage. *A point of interest $i \in \mathcal{I}$ is said to be* covered *by a configuration $\mathbf{q} \in \mathcal{X}$ or by a path P if $i \in \mathcal{S}(\mathbf{q})$ or if $i \in \mathcal{S}(P)$, respectively. In such a setting, we say that $\mathbf{q}$ (or P) covers the point of interest i.*

Given a start configuration $\mathbf{q}_{\text{s}} \in \mathcal{X}$, POI $\mathcal{I}$, and a sensor model $\mathcal{S}$, the inspection planning problem calls for computing a collision-free path P starting at $\mathbf{q}_{\text{s}}$ which maximizes $|\mathcal{S}(P)|$ while minimizing $\ell(P)$. Note that this is *not* a bicriteria optimization problem—our primary optimization function is maximizing the coverage of our path. Out of all such paths we are interested in the shortest one.

4 Method Overview

In this section, we provide an overview of IRIS—our algorithmic framework for computing asymptotically optimal inspection plans. A key algorithmic tool in our approach is to cast the continuous inspection planning problem (Sec. 3) to a discrete version of the problem where we only consider a finite number of configurations from which we inspect the POI, and a discrete set of feasible movements between those configurations. The assumption that inspection of POI only happens at vertices is not significantly limiting since a robot's motion can be approximated by multiple vertices, and many inspection applications require non-zero time to complete a high-quality sensor measurement (e.g., to obtain non-blurry images, high-accuracy lidar scans, etc.), which can be effectively encoded at vertices. Thus, we start in Sec. 4.1 by formally defining the *graph inspection problem* and then continue in Sec. 4.2 to provide an overview of how IRIS builds and uses such graphs. We then describe the method in detail in Sec. 5, and

in Sec. 6 show that IRIS's solution converges to the length and coverage of an optimal inspection path.

4.1 Graph inspection problem

Similar to the (continuous) inspection problem, a *graph inspection problem* is a tuple $(\mathcal{G}, \mathcal{I}, \mathcal{S}, \ell, v_{\mathrm{s}})$ where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a motion-planning roadmap (namely, a graph embedded in $\mathcal{X}$, in which every vertex $v \in \mathcal{V}$ is a configuration and every edge $(u, v) \in \mathcal{E}$ denotes the transition from configuration u to v), $\mathcal{I}$ and $\mathcal{S}$ are defined as in Sec. 3, $\ell : \mathcal{E} \to \mathbb{R}$ denotes the length of each edge in the roadmap, and v_{s} is the start vertex (corresponding to the start configuration $\mathbf{q}_{\mathrm{s}}$). A path P on $\mathcal{G}$ is represented by a sequence of vertices $v_i \in \mathcal{V}$ such that $P = \{v_0, \ldots, v_{n-1}\}$, $v_0 = v_s$ and $(v_i, v_{i+1}) \in \mathcal{E}$. It is important to note that there can be loops in a path, so it is possible that $v_m = v_k$ for $m \neq k$. The *length* and *coverage* of P are defined as the total length of P's edges and the set of all points inspected when traversing P, respectively. Namely, $\ell(P) := \sum_{i=0}^{n-2} \ell(v_i, v_{i+1})$ and $\mathcal{S}(P) := \bigcup_{v \in P} \mathcal{S}(v)$. The *optimal* graph inspection problem calls for a path P^* that starts at v_{s} and maximizes the number of points inspected. Out of all such paths, P^* has the minimal length. Finally, a path is said to be *near-optimal* for some $\varepsilon \geq 0$ and $p \in (0, 1]$ if $|\mathcal{S}(P)|/|\mathcal{S}(P^*)| \geq p$ and $\ell(P) \leq (1 + \varepsilon) \cdot \ell(P^*)$.

4.2 Overview of IRIS

Our algorithmic framework, depicted in Fig. 2, interleaves sampling-based motion planning and graph search. Specifically, we incrementally construct an RRT $\mathcal{T}$ rooted at $\mathbf{q}_{\mathrm{s}}$ which implicitly defines a corresponding RRG $\mathcal{G}$. All edges in $\mathcal{T}$ are checked for collision with the environment during its construction (so the roadmap is guaranteed to be connected) while all the other edges of $\mathcal{G}$ are not explicitly checked for collision. Lazy edge evaluation, common in motion-planning algorithms (Bohlin and Kavraki 2000; Hauser 2015; Dellin and Srinivasa 2016; Salzman and Halperin 2015), allows us to defer collision detection until absolutely necessary and reduce computational effort. This is critical in our domain of interest where computing $\mathrm{Shape}(\cdot)$, the geometry of the robot as is defined in Sec. 3, typically dominates algorithms' running times (Niyaz et al. 2018).

The roadmap $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ induces the subset of the POI that can be inspected, denoted as $\mathcal{I}_{\mathcal{G}} := \bigcup_{v \in \mathcal{V}} \mathcal{S}(v)$. Given two approximation parameters $\varepsilon \geq 0$ and $p \in (0, 1]$, we compute a near-optimal inspection path for the graph-inspection problem $(\mathcal{G}, \mathcal{I}_{\mathcal{G}}, \mathcal{S}, \ell, v_{\mathrm{s}})$ by casting the problem as a graph-search problem on a different graph $\mathcal{G}_{\mathcal{S}}$ (to be defined shortly).

As we add vertices and edges to $\mathcal{T}$ incrementally, the roadmap $\mathcal{G}$ is incrementally densified. In addition, we tighten approximations by decreasing ε and increasing p between iterations. As we will see (Sec. 6), this will ensure that IRIS is asymptotically optimal.

5 Method

In this section, we detail the different components of IRIS. Sec. 5.1 and 5.2 describe how we construct a roadmap and then search it, respectively. After describing in Sec 5.3 how we modify the approximation parameters used by IRIS, we conclude in Sec. 5.4 with implementation details.

5.1 Roadmap construction

We construct a sequence of graphs embedded in $\mathcal{X}$. Specifically, denote the RRT constructed at the i'th iteration as $\mathcal{T}_i$ defined over the set of vertices $\mathcal{V}_i$. We start with an empty tree rooted at $\mathbf{q}_{\mathrm{s}}$ and at the i'th iteration sample a random configuration, compute its nearest neighbor in $\mathcal{T}_i$, and extend that vertex toward the random configuration. If that extension is collision-free we add it to the tree. If not, we repeat this process (see Kuffner and LaValle (2000); LaValle (2006) for additional details regarding RRT).

Note that it is not necessary to add only one collision-free configuration in each roadmap update. Adding multiple configurations in one iteration does not hurt the theoretical guarantees while providing us more room to improve an inspection plan in terms of both inspection and path length. There are different possible strategies to balance between roadmap construction and graph search on the roadmap. One example is Fu et al. (2021) where a condition on additional inspection coverage is used to trigger graph searches on the updated roadmap. As the major focus of this paper is to provide a theoretical foundation for the proposed algorithm framework, we only discuss the variant where configurations are added one at a time.

The tree $\mathcal{T}_i$ implicitly defines an RRG $G_i = (\mathcal{V}_i, \mathcal{E}_i)$ defined over the same set of vertices where every vertex is connected to all other vertices within distance r_i. Here, we define r_i as in Lemma 3 which will allow us to prove that our approach is asymptotically optimal (see Sec. 6).

5.2 Graph inspection planning

We use the RRG described in Sec. 5.1 to define a graph inspection problem, and then compute near-optimal inspection paths over this graph. Before describing how we compute near-optimal inspection paths, we first describe how we compute optimal paths given a graph inspection problem.

5.2.1 Optimal planning Given a graph inspection problem $(\mathcal{G}, \mathcal{I}_{\mathcal{G}}, \mathcal{S}, \ell, v_{\mathrm{s}})$, we compute optimal inspection paths by formulating our inspection problem as a graph-search problem on an *inspection graph* $\mathcal{G}_{\mathcal{S}} := (\mathcal{V}_{\mathcal{S}}, \mathcal{E}_{\mathcal{S}})$. Here, vertices are pairs comprised of a vertex $u \in \mathcal{V}$ in the original graph and subsets of $\mathcal{I}_{\mathcal{G}}$. Namely, $\mathcal{V}_{\mathcal{S}} = \mathcal{V} \times 2^{\mathcal{I}_{\mathcal{G}}}$, and note that $|\mathcal{V}_{\mathcal{S}}| = O\left(|\mathcal{V}| \cdot 2^{|\mathcal{I}_{\mathcal{G}}|}\right)$. An edge e between vertices $(u, \mathcal{I}_u)$ and $(v, \mathcal{I}_v)$ exists if $(u, v) \in \mathcal{E}$ and $\mathcal{I}_u \cup \mathcal{S}(v) = \mathcal{I}_v$. If it exists, its cost is simply $\ell(u, v)$.

Any (possibly non-simple) path P_G in the original graph G from v_{s} to u can be represented by a corresponding path $P_{\mathcal{G}_{\mathcal{S}}}$ in the inspection graph $\mathcal{G}_{\mathcal{S}}$, from $(v_{\mathrm{s}}, \mathcal{S}(v_{\mathrm{s}})) \in \mathcal{V}_{\mathcal{S}}$ to $(u, \mathcal{S}(P_G)) \in \mathcal{V}_{\mathcal{S}}$, and $\ell(P_G) = \ell(P_{\mathcal{G}_{\mathcal{S}}})$. Thus, our algorithm will run an A*-like search from $(v_{\mathrm{s}}, \mathcal{S}(v_{\mathrm{s}})) \in \mathcal{V}_{\mathcal{S}}$ to any vertex in the goal set $\mathcal{V}_{\mathrm{goal}} = \{(v, \mathcal{I}_{\mathcal{G}}) | v \in \mathcal{V}\}$. An optimal inspection path is the shortest path between $(v_{\mathrm{s}}, \mathcal{S}(v_{\mathrm{s}}))$ and any vertex in $\mathcal{V}_{\mathrm{goal}}$. For a visualization, see Fig. 3. Note that here we assume the graph $\mathcal{G}$ is connected and that the set of points to be inspected is $\mathcal{I}_{\mathcal{G}}$. This implies that an optimal inspection path always exists.

We can speed up this naïve algorithm using the notion of *dominance*, which is used in many shortest-path

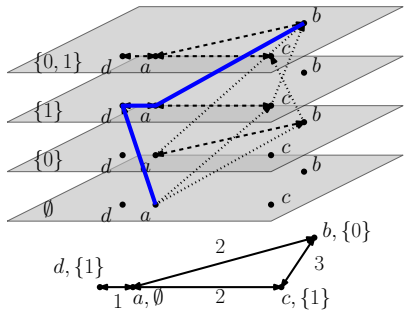


Figure 3. Computing optimal inspection paths on graphs by casting a graph-inspection problem (bottom) to a graph-search problem (top). Grey layers correspond to the set of all vertices in $\mathcal{V}_\mathcal{S}$ that share the same set of points inspected. Edges connecting vertices in the same (different) layer are depicted in dashed (dotted) lines, respectively. The start is $(a, \emptyset)$ and the goal set $\mathcal{V}_{\text{goal}}$ contains all vertices in the top layer. Notice that the optimal path (blue) visits vertex a twice.

algorithms (see, e.g., Salzman et al. (2017)). In our context, given two paths P, P' in our original roadmap $\mathcal{G}$ that start and end at the same vertices, we say that P *dominates* P' if $\ell(P) \leq \ell(P')$ and $\mathcal{S}(P) \supseteq \mathcal{S}(P')$. Clearly, P is always preferred over P'. Thus, when searching $\mathcal{G}_\mathcal{S}$, if we compute a shortest path to some node $(u, \mathcal{I}_u)$ of length ℓ_u, we do not need to consider any path of length larger than ℓ_u from all vertices $(u, \mathcal{I}'_u)$ such that $\mathcal{I}'_u \subseteq \mathcal{I}_u$. For pseudo-code describing a general A*-like search algorithm to optimally solve the graph-inspection problem, see Alg. 1 *without* lines 17-27.

While path domination may prune away paths in the open list of the A*-like search, this algorithm is hardly practical due to the exponential size of the search space (recall that $|\mathcal{V}_\mathcal{S}| = O(|\mathcal{V}_\mathcal{S}| \cdot 2^{|\mathcal{I}_\mathcal{G}|})$). In the next sections, we show how to prune away large portions of the search space by extending the notion of dominance to *approximate dominance*.

5.2.2 Near-optimal planning To perform near-optimal planning, we introduce the idea of approximate dominance. Approximate dominance is a relaxed version of the (strong) dominance mentioned above, characterized by approximation parameters.

Let P, P' be two paths in $\mathcal{G}$ that start and end at the same vertices and let $\varepsilon \geq 0$ and $p \in (0, 1]$ be some approximation parameters.

Definition 2. ε, p-domination. *We say that path P ε, p-dominates path P' if $\ell(P) \leq (1+\varepsilon) \cdot \ell(P')$ and $|\mathcal{S}(P)| \geq p \cdot |\mathcal{S}(P) \cup \mathcal{S}(P')| = p \cdot |\mathcal{I}_\mathcal{G}|$.*

Note that it is possible that both P ε, p-dominates P' and P' ε, p-dominates P. For a visualization of the notions of dominance and the approximate dominance, see Fig. 4.

Intuitively, approximate dominance allows us to dramatically prune the search space by only considering paths that can significantly improve the quality (either in terms of length or the set of POI inspected) of a given path. When pruning away (strongly) dominated paths, it is clear that they cannot be useful in the future. However, if we prune away approximate-dominated paths, we need to efficiently account for all paths that were pruned away in order to bound the quality of the solution obtained. If we prune away approximate-dominated paths without any special consideration, the "errors" introduced by each domination accumulate. To bound the accumulated "error", during the search, we need to consistently keep track of "what is the best inspection path if we do not perform approximate dominations that happened so far"? Getting the best inspection paths precisely is equivalent to optimal planning. Thus, we use estimations and name such estimations *potentially achievable paths* or PAP s.

Definition 3. Potentially achievable path. *A potentially achievable path (PAP) $\tilde{P}$ to some vertex $u \in \mathcal{V}$ is a pair $(\tilde{\ell}, \tilde{\mathcal{I}})$ such that $\tilde{\ell} \geq 0$ and $\mathcal{S}(u) \subseteq \tilde{\mathcal{I}} \subseteq \mathcal{I}_\mathcal{G}$. By a slight abuse of notation, we extend the definitions of $\ell(\cdot)$ and $\mathcal{S}(\cdot)$ such that $\ell(\tilde{P}) = \tilde{\ell}$ and $\mathcal{S}(\tilde{P}) = \tilde{\mathcal{I}}$.*

It may seem that a PAP is simply a path but note (as the name PAP suggests) that we do not require that there exists any path P from v_s to u such that $\ell(P) = \ell(\tilde{P})$ and $\mathcal{S}(P) = \mathcal{S}(\tilde{P})$. It merely states that such a path *may* exist.

We now use PAP s to define the notion of a *path pair*:

Definition 4. Path pair. *Let P and $\tilde{P}$ be a path and a PAP from v_s to some $v \in \mathcal{V}$ such that $\ell(\tilde{P}) \leq \ell(P)$ and $\mathcal{S}(\tilde{P}) \supseteq \mathcal{S}(P)$. Their path pair is PP $:= (P, \tilde{P})$ and we call P and $\tilde{P}$ the achievable and potentially achievable paths of PP, respectively.*

Let us define operations on PAP s and on PP s, visualized in Fig. 5. The first operation we consider is *extending* a PAP $\tilde{P}_u$ by an edge $e = (u, v)$, denoted as $\tilde{P}_u + e$. This can be thought of as appending e to $\tilde{P}_u$, had it existed and thus accounting for the length $\ell(e)$ and additional coverage $\mathcal{S}(v)$. Formally, extending $\tilde{P}_u + e$ yields a PAP $\tilde{P}_v$ such that $\ell(\tilde{P}_u) = \ell(\tilde{P}_v) + \ell(e)$ and $\mathcal{S}(\tilde{P}_u) = \mathcal{S}(\tilde{P}_v) \cup \mathcal{S}(u)$. Extending the path pair $\text{PP}_u = (P_u, \tilde{P}_u)$ by the edge $e = (u, v)$ (denoted as $\text{PP}_u + e$) simply extends both P_u and $\tilde{P}_u$ by e. This yields the path pair $\text{PP}_v = (P_v, \tilde{P}_v)$ where $\ell(P_v) = \ell(P_u) + \ell(e)$, $\mathcal{S}(P_v) = \mathcal{S}(P_u) \cup \mathcal{S}(v)$ and $\tilde{P}_v = \tilde{P}_u + e$.

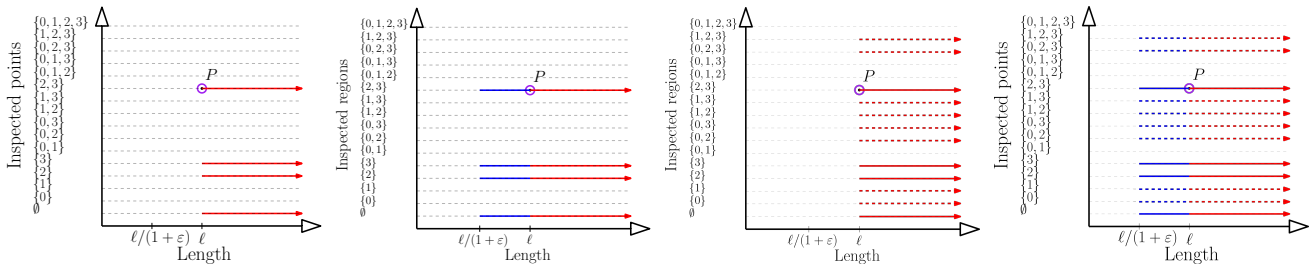
The second operation is *subsuming* a path pair by another one which can be thought of as constructing a PAP that dominates the PAPs of both path pairs. Formally, Let $\text{PP}_1 = (P_1, \tilde{P}_1)$ and $\text{PP}_2 = (P_2, \tilde{P}_2)$ be two path pairs such that both connect the start vertex v_s to some vertex $u \in \mathcal{V}$. The path pair defined by PP_1 subsuming PP_2 is

$$\text{PP}_1 \oplus \text{PP}_2 := (P_1, (\min\{\ell(\tilde{P}_1), \ell(\tilde{P}_2)\}, \mathcal{S}(\tilde{P}_1) \cup \mathcal{S}(\tilde{P}_2))).$$

We now define the notion of bounding a path pair which will be crucial for ensuring near-optimal solutions:

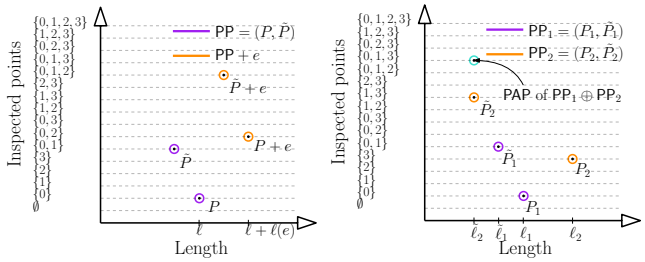
Definition 5. ε, p-bounded. *A path pair PP $:= (P, \tilde{P})$ is said to be ε, p-*bounded *for some $\varepsilon \geq 0$ and $p \in (0, 1]$ if P ε, p-dominates $\tilde{P}$.*

To compute a near-optimal inspection path (Alg. 1 and Fig. 6), we extend each path considered by our search algorithm to be a path pair and use this additional data to prune away approximately dominated paths. Similar to A*, our algorithm uses two priority queues OPEN and CLOSED to track the nodes considered by the search. It starts by inserting the start vertex $(v_\text{s}, \mathcal{S}(v_\text{s}))$ to the OPEN list together with the path pair $\text{PP}_s = (P_\text{s}, P_\text{s})$ (here P_s is a path containing only start vertex v_s) (line 2).



(a) Dominance **(b)** ε-dominance **(c)** p-dominance **(d)** (ε, p)-dominance

Figure 4. Visualization of the notion of dominating paths by considering a path P from v_s to some vertex u as a two-dimensional point $(\ell(P), \mathcal{S}(P))$. Here $\mathcal{I}_\mathcal{G} = \{0, 1, 2, 3\}$ and P is depicted using the purple circle with $\ell(P) = \ell$ and $\mathcal{S}(P) = \{2, 3\}$. All paths from v_s to u that (a) are dominated by P (solid red), (b) are ε-dominated by P (solid blue), (c) are p-dominated by P for $p = 60\%$ (dashed red), (d) are ε, p-dominated by P for $\varepsilon > 0$ and $p = 60\%$ (dashed blue).



(a) Extending a path pair **(b)** Subsuming a path pair

Figure 5. Depiction of operations on path pairs. (a) PP extended by an edge $e = (u, v)$ with $\mathcal{S}(v) = \{2\}$. (b) PP_1 subsuming PP_2. Note that P_1 is the achievable path of $\mathsf{PP}_1 \oplus \mathsf{PP}_2$ thus only the potentially achievable path is explicitly marked.

Our algorithm proceeds in a similar fashion to A*—we pop the most promising node $n = (u, \mathcal{I}_u, \mathsf{PP}_u)$ from OPEN (line 4) and move it to CLOSED (line 5). If the PAP of this node is in the goal set $\mathcal{V}_\mathrm{goal}$ (line 6), we terminate the search and return the achievable path of PP_u (line 7). If not, we consider all neighboring edges e of u in $\mathcal{G}$ and extend the node n (line 9). This is akin to computing n's neighbors in $\mathcal{G}_\mathcal{S}$.

At this point, our algorithm deviates from the standard A* algorithm. For each newly created node $(v, \mathcal{I}_v, \mathsf{PP}_v)$ we check if there exists a node in CLOSED whose PAP dominates $\tilde{P}_v$. If so, this node is discarded (lines 11-14). If no such node exists in the CLOSED list, we check if there exists a node in OPEN that may subsume it. If so, that node is updated and this node is discarded (line 17-21). Finally, we check if this node can subsume nodes that are in OPEN. If so, such nodes are discarded and this node is updated. (line 24-27).

We prove that, Alg. 1 returns a path that ε, p-dominates an optimal inspection path on the roadmap (see Sec. 6).

Algorithm 1 Near-optimal inspection graph search
Input: $(\mathcal{G}_\mathcal{S}, v_\mathrm{s}, \mathcal{V}_\mathrm{goal}, \varepsilon, p)$

1: CLOSED $\leftarrow \emptyset$
2: OPEN $\leftarrow (v_\mathrm{s}, \mathcal{S}(v_\mathrm{s}), \mathsf{PP}_\mathrm{s})$
3: **while** OPEN $\neq \emptyset$ **do**
4: $(u, \mathcal{I}_u, \mathsf{PP}_u) \leftarrow$ OPEN.extract_min()
5: CLOSED.insert$(u, \mathcal{I}_u, \mathsf{PP}_u)$
6: **if** $\tilde{P}_u \in \mathcal{V}_\mathrm{goal}$ **then** ▷ $\tilde{P}_u$ is the PAP of PP_u
7: **return** P_u ▷ P_u is the achievable path of PP_u
8: **for** $e = (u, v) \in$ neighbors$(u, \mathcal{G})$ **do**
9: $(v, \mathcal{I}_v, \mathsf{PP}_v) \leftarrow$ extend$((u, \mathcal{I}_u, \mathsf{PP}_u), e)$
10: valid = **True**
11: **for** $(v, \mathcal{I}'_v, \mathsf{PP}'_v) \in$ CLOSED **do**
12: **if** $\tilde{P}'_v$ dominates $\tilde{P}_v$ **then**
13: valid = **False**
14: **break**
15: **if** !valid **then**
16: **continue**
17: **for** $(v, \mathcal{I}'_v, \mathsf{PP}'_v) \in$ OPEN **do**
18: **if** $\mathsf{PP}'_v \oplus \mathsf{PP}_v$ is ε, p-bounded **then**
19: $(v, \mathcal{I}'_v, \mathsf{PP}'_v) \leftarrow (v, \mathcal{I}'_v, \mathsf{PP}'_v \oplus \mathsf{PP}_v)$
20: valid = **False**
21: **break**
22: **if** !valid **then**
23: **continue**
24: **for** $(v, \mathcal{I}'_v, \mathsf{PP}'_v) \in$ OPEN **do**
25: **if** $\mathsf{PP}_v \oplus \mathsf{PP}'_v$ is ε, p-bounded **then**
26: OPEN.remove$(v, \mathcal{I}'_v, \mathsf{PP}'_v)$
27: $(v, \mathcal{I}, \mathsf{PP}_v) \leftarrow (v, \mathcal{I}, \mathsf{PP}_v \oplus \mathsf{PP}'_v)$
28: OPEN$\leftarrow (v, \mathcal{I}_v, \mathsf{PP}_v)$
29: **return** NULL

5.3 Tightening approximation factors

Recall that our algorithm starts with approximation parameters p_0 and ε_0. We endow our algorithm with a tightening factor $f \in (0, 1]$, and at the i'th iteration we set our approximation parameters as $p_i = p_{i-1} + f \cdot (1 - p_{i-1})$ and $\varepsilon_i = \varepsilon_{i-1} + f \cdot (0 - \varepsilon_{i-1})$. As we will see (Sec. 6), since $\lim_{i\to\infty} p_i = 1$, $\lim_{i\to\infty} \varepsilon_i = 0$, the tightening allows our method to achieve asymptotic optimality.

5.4 Implementation details

5.4.1 Lazy computation in graph inspection planning We run our search algorithm on $\mathcal{G}$ (Alg. 1) without checking if its edges are collision-free or not (recall that only the edges of $\mathcal{T}$ were explicitly checked for collision). Once a solution is found, we start checking edges one by one until the entire path was found to be collision-free or until one edge is found to be in collision, in which case we remove it from the edge

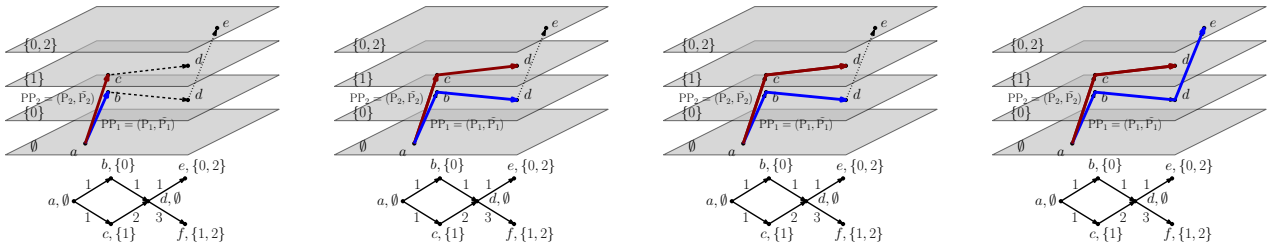


Figure 6. Visualization of Alg. 1 initialized with $\varepsilon = 2/3$ and $p = 1/2$ (only the relevant parts of the inspection graph are depicted). The search starts from $(a, \emptyset)$ with the trivial PAP of length zero and no points inspected. (a) Two paths (red and blue) are extended from the start node to $(b, \{0\})$ and $(c, \{1\})$ with path pairs PP_1 and PP_2, respectively (the PAP s of each path have the same length and coverage as the paths themselves). (b) Blue path extended to $(d, \{0\})$ with $\ell(P_1) = \ell(\tilde{P}_1) = 2$ and $\mathcal{S}(P_1) = S(\tilde{P}_1) = \{0\}$. (c) Red path extended to $(d, \{1\})$ with $\ell(P_2) = \ell(\tilde{P}_2) = 3$ and $\mathcal{S}(P_2) = S(\tilde{P}_2) = \{1\}$. Here, $\mathsf{PP}_1 \oplus \mathsf{PP}_2$ ε, p-dominates PP_2 and the red path is discarded and PAP_1 is updated to have length 2 and coverage $\{0, 1\}$ (d) Blue path extended to vertex $(e, \{0, 2\})$. Here, $\ell(P_1) = \ell(\tilde{P}_1) = 3$ and $\mathcal{S}(P_1) = \{0, 2\}, S(\tilde{P}_1) = \{0, 1, 2\}$. The algorithm terminates with the path $a - b - d - e$ whose length is 3 and has coverage of $\{0, 2\}$. Notice that the path $a - c - d - e$ (not computed) is optimal as its length is four and it has complete coverage. The computed path is within the bounds ensured by the approximation factor p and ε.

set. This approach is common to speed up motion-planning algorithms when edges are expensive to evaluate (Dellin and Srinivasa 2016; Haghtalab et al. 2018).

5.4.2 Node extension in graph inspection planning Any optimal inspection path can be decomposed into a sequence of vertices that improve the coverage of the path called *milestones*. It is straightforward to see that an optimal inspection path will (i) terminate at a milestone and (ii) connect a pair of milestones via a shortest path in $\mathcal{G}$. Following this observation, instead of extending each path P from a vertex u by all outgoing edges in $\mathcal{G}$ (Alg. 1, line 8), we run a Dijkstra-like search from u and collect all first-met vertices that can be milestones.

5.4.3 Heuristic computation in graph inspection planning Recall that A* orders nodes in the OPEN list according to their computed cost-to-come added to a heuristic estimate of their cost to reach the goal. The heuristic function that we use for vertex $(u, \mathcal{I}_u)$ is computed as follows: we run a Dijkstra search on $\mathcal{G}$ from u and consider the vertices $\mathcal{V}_u$ encountered during the search. We terminate when $\left(\bigcup_{v \in \mathcal{V}_u} \mathcal{S}(v)\right) \cup \mathcal{I}_u = \mathcal{I}_G$ and use the maximal distance between u to any node in $\mathcal{V}_u$ as our admissible (Hart et al. 1968) heuristic function. We now show why the heuristic is admissible. When we terminate with $\left(\bigcup_{v \in \mathcal{V}_u} \mathcal{S}(v)\right) \cup \mathcal{I}_u = \mathcal{I}_G$, denote the last vertex added to $\mathcal{V}_u$ as v_{last}. Note that v_{last} is also the most distant (via roadmap edges) vertex from u due to the nature of Dijkstra's algorithm. According to the termination condition, there exists a non-empty set of POI $\mathcal{I}_{\text{last}}$ that is covered by v_{last} but not covered by any other vertices in $\mathcal{V}_u$. Namely, $\mathcal{I}_{\text{last}} \subset \mathcal{S}(v_{\text{last}})$ and $\mathcal{I}_{\text{last}} \cap \left(\bigcup_{v \in \mathcal{V}_u, v \neq v_{\text{last}}} \mathcal{S}(v)\right) = \emptyset$. To achieve an inspection coverage of $\mathcal{I}_{\mathcal{G}}$, the minimum distance to travel is equal to or greater than the distance between u and v_{last} because with any shorter path, $\mathcal{I}_{\text{last}}$ is not covered.

6 Theoretical Guarantees

In this section, we provide theoretical properties showing that IRIS is guaranteed to be asymptotically optimal, given that the system and the environment satisfy several general assumptions. For brevity, we only state the main definitions, lemmas, and theorems and defer all proofs to Appendix A and Appendix B. Recall that IRIS iteratively densifies the roadmap and runs a near-optimal inspection graph search on the latest roadmap. Thus, we begin (Thm. 1) by proving that our graph-inspection search algorithm (Alg. 1) returns a near-optimal solution when compared to the optimal solution (available for that specific roadmap). We then continue (Thm. 2) to show that IRIS is asymptotically optimal when the approximation parameters satisfy that $\lim_{N \to \infty} \varepsilon_N = 0$ and $\lim_{N \to \infty} p_N = 1$, where N is the number of vertices in the roadmap. This is done by using the notion of *probablistic exhaustivity* (Schmerling et al. 2015) coupled together with the assumptions that (i) an optimal inspection trajectory is *well behaved* (Def. 8) (ii) an optimal inspection trajectory *weak δ_{cl}-clearance* and that (iii) the inspection problem is *regular* (Def. 10). It is worth noting that all proofs are general enough to account for complex robotic systems, including those having differential constraints. Having said that, as we use a graph-based search method, we require a method is required to compute valid motions (if such motions exist) between two close-by configurations.

We start by proving that Alg. 1 is near-optimal.

Definition 6. Optimal inspection path on a roadmap. *Let $(\mathcal{G}, \mathcal{I}_{\mathcal{G}}, \mathcal{S}, \ell, v_{\text{s}})$ be a graph inspection problem. An* optimal inspection path *P^* is a path on roadmap $\mathcal{G}$, starting at v_{s}, and satisfies*

$$\ell(P^*) = \operatorname{argmin}\{\ell(P) | P \text{ is a path with } \mathcal{S}(P) = \mathcal{I}_{\mathcal{G}}\}.$$

Where $\ell^* := \ell(P^*)$ and $\mathcal{S}^* := \mathcal{S}(P^*) = \mathcal{I}_{\mathcal{G}}$, denote the length and coverage of an optimal path, respectively.

Lemma 1. *Alg. 1 (near-optimal inspection graph search), all path pairs in OPEN and CLOSED during the search are ε, p-bounded.*

Lemma 2. *Let $P^* = \{v_0, v_1, \ldots, v_{n^*}\}$ be an optimal inspection path and denote $P^*[i] := \{v_0, \ldots, v_i\}$, for $i \in [0, n^*]$ as the path composed of the first i waypoints of P^*. During every iteration of Alg. 1, there exists a path pair $\mathsf{PP}_v = (P_v, \tilde{P}_v)$ in OPEN and an index i such that $v = v_i$ and $\tilde{P}_v$ strictly dominates $P^*[i]$. Namely, $\ell(\tilde{P}) \leq \ell(P^*[i])$ and $\mathcal{S}(P^*[i]) \subseteq \mathcal{S}(\tilde{P})$.*

With Lemma 1 and 2, we can show that Alg. 1 returns a near-optimal result.

Theorem 1. Near-optimal inspection graph search. *Near-optimal inspection graph search (Alg. 1) computes a path P that ε, p-dominates an optimal inspection path P^*. Namely, $\ell(P) \leq (1+\varepsilon) \cdot \ell(P^*)$ and $|\mathcal{S}(P)| \geq p \cdot |\mathcal{S}(P^*)|$.*

We now continue to prove that IRIS is asymptotically optimal. To prove this, we show that an optimal inspection path x^* can be approximated by a sequence of configurations sampled by IRIS, given certain conditions. Specifically, we will need to show that as the number of iterations approaches infinity, the following requirements hold: (i) the length of the path induced by this sequence of samples converges asymptotically to the length of x^*, (ii) the coverage obtained by this sequence of samples converges asymptotically to the coverage of x^* and that (iii) our inspection graph search algorithm finds such a sequence of samples. To do so, we rely on the notion of *probabilistic exhaustivity* (Schmerling et al. 2015). Roughly speaking, it is the notion that given a sufficiently large set of uniformly sampled configurations, any path can be traced arbitrarily well in the configuration space by a path defined as a sequence of configurations.

Lemma 3. Probabilistic exhausitivity. *Let $x : [0,T] \to \mathcal{X}_{\text{free}}$ be a dynamically feasible trajectory. Let $\mathcal{Q}_N$ be a set of N points sampled independently and identically from the uniform distribution on the collision-free space $\mathcal{X}_{\text{free}}$ and set $V_N = \{x(0)\} \cup \mathcal{Q}_N$. For a given N, set*

$$r_N = \kappa \cdot (\log(N)/N)^{1/D}.$$

Here, D is a constant capturing the dimension of the system and κ is a commutable constant depending on the system dynamics, N, D, and some tuning parameter $\eta \geq 0$. Let $\tilde{\mathcal{A}}_N$ be the event that there exists a discrete sequence of configurations $P = \{\mathbf{q}_i\}_{i=1}^n \subseteq V_N$ that (δ, r)-traces x for any $\delta \in (0,1)$ and $r = r_N$. The probability that event $\tilde{\mathcal{A}}_N$ doesn't happen, denoted by $\mathbb{P}(\tilde{\mathcal{A}}_N^c)$ is asymptotically bounded by

$$\mathbb{P}(\tilde{\mathcal{A}}_N^c) \leq O\left(N^{-\eta} \log^{-\frac{1}{D}} N\right).$$

As is defined above, $\eta \geq 0$ is some tuning parameter.

In addition, we assume that x^* is *well-behaved*. Roughly speaking, this ensures that there are no singular points along the trajectory where a POI can only be inspected from. This, in turn, will allow us to ensure that trajectories that trace an optimal inspection path cover the same set of POI.

Definition 7. Inspecting configuration region. *Let $i \in \mathcal{I}$ be a point of interest (POI), the inspecting configuration region of i, denoted as $\mathcal{X}_{\text{insp}}(i)$, is defined to be the union of all configurations from which the POI can be inspected. Namely,*

$$\mathcal{X}_{\text{insp}}(i) = \{\mathbf{q} \in \mathcal{X}_{\text{free}} : i \in \mathcal{S}(\mathbf{q})\}.$$

Similarly, the inspecting configuration region of $\mathcal{I}' \subseteq \mathcal{I}$ is defined as

$$\mathcal{X}_{\text{insp}}(\mathcal{I}') = \{\mathbf{q} \in \mathcal{X}_{\text{free}} : \mathcal{I}' \subseteq \mathcal{S}(\mathbf{q})\}.$$

Definition 8. Well-behaving of an inspection trajectory. *Let $x : [0,T] \to \mathcal{X}_{\text{free}}$ be a feasible inspection trajectory. x is said to be strongly ξ-well behaved if $\forall i \in \mathcal{S}(x)$, there exist at least one point along x whose ξ-neighborhood is completely within the inspecting configuration region of i. Namely,*

$$\forall i \in \mathcal{S}(x), \exists t \in [0,T] \text{ s.t. } B^e(x(t), \xi) \subseteq \mathcal{X}_{\text{insp}}(i).$$

Similarly, x is said to be weakly ξ-well behaved if $\forall i \in \mathcal{S}(x) / (\mathcal{S}(x(0)) \cup \mathcal{S}(x(T)))$, there exists at least one point along x whose ξ-neighborhood is completely within the inspecting configuration region of i. Namely,

$$\forall i \in \mathcal{S}(x) / (\mathcal{S}(x(0)) \cup \mathcal{S}(x(T))), \exists t \in [0,T] \text{ s.t. } B^e(x(t), \xi) \subseteq \mathcal{X}_{\text{insp}}(i).$$

It is not hard to see, that any strongly well-behaved trajectory can be shortened to a weakly well-behaved trajectory without loosing coverage. Thus an optimal inspection trajectory can only be weakly well-behaved.

We further require the inspection-planning problem to be *regular*. The notion of regularity is required because for a weakly well-behaved optimal inspection trajectory $x^* : [0,T] \to \mathcal{X}_{\text{free}}$ we need to take special care to cover the POI inspected at $x^*(T)$. The notion of regularity will ensure that there always exists a region near $x^*(T)$ that IRIS can sample inside.

Definition 9. Regular boundary. *A set $\mathcal{X}' \subseteq \mathcal{X}_{\text{free}}$ is said to have a regular boundary if there exists $\gamma > 0$ such that $\forall \mathbf{q} \in \partial\mathcal{X}$, there exists $\mathbf{q}' \in \mathcal{X}$ with $B^e(\mathbf{q}', \gamma) \subseteq \mathcal{X}'$ and $\mathbf{q} \in \partial B^e(\mathbf{q}', \gamma)$.*

Definition 10. Regularity of an inspection-planning problem. *Let $\mathcal{P} = (\mathcal{X}, \mathcal{I}, \mathcal{S}, \ell, q_s)$ be an inspection-planning problem and $\mathcal{X}_{\text{free}}$ be the set of collision-free configurations. $\mathcal{P}$ is said to be regular if $\forall \mathbf{q} \in \mathcal{X}_{\text{free}}$, $\mathcal{X}_{\text{insp}}(\mathcal{S}(\mathbf{q}))$ has a regular boundary.*

Similar to many other analyses of sampling-based planning algorithms (see, e.g., Kavraki et al. (1996); Karaman and Frazzoli (2011); Solovey et al. (2018)), we also assume that an optimal trajectory to trace has clearance from $\mathcal{X}_{\text{obs}}$.

Definition 11. Strong/weak δ_{cl}-clearance. *Let $x : [0,T] \to \mathcal{X}_{\text{free}}$ be a feasible trajectory. x has strong δ_{cl}-clearance if $\forall t \in [0,T]$, $x(t)$ is in δ_{cl}-interior of $\mathcal{X}_{\text{free}}$ (namely, $x(t)$ is at least δ_{cl} away from any point in $\mathcal{X}_{\text{obs}}$ using the Euclidean distance). Furthermore, x has weak δ_{cl}-clearance if there exists a sequence of homotopic paths $\{x_k\}_{k\in\mathbb{N}}$ that satisfies:*

(i) $\lim_{k\to\infty} x_k = x$.

(ii) x_0 *has strong* δ_{cl}*-clearance.*

(iii) $\forall k \in [0,\infty)$, x_k *is dynamically feasible and has strong* δ_k*-clearance for some* $\delta_k > 0$*, and* $\lim_{k\to\infty} \delta_k = 0$.

(iv) $\lim_{k\to\infty} \ell(x_k) = \ell(x)$.

We are finally ready to state our final theorem.

Theorem 2. IRIS asymptotic optimality. *Let $\mathcal{P} = (\mathcal{X}, \mathcal{I}, \mathcal{S}, \ell, \mathbf{q}_s)$ be a regular inspection-planning problem and $\mathcal{X}_{\text{free}}$ be the collision-free space. Assume that the robot*

system satisfies the assumptions mentioned in Schmerling et al. (2015) and Let $x^* : [0, T] \to \mathcal{X}_{\text{free}}$ *be an optimal feasible inspection trajectory such that*

(i) $x^*(0) = \mathbf{q}_s$,

(ii) x^* *has weak* δ_{cl}*-clearance for some* $\delta_{\text{cl}} > 0$,

(iii) x^* *is weakly* ξ*-well behaved for some* $\xi > 0$.

Furthermore, let ℓ_i *and* $\mathcal{S}_i$ *denote the arc length, and the inspection coverage, respectively of the trajectory returned by* IRIS *at the* N*'th iteration using approximation parameters* ε_N *and* p_N*, and a lower-bound connecting radius* r_N *(as defined in Lemma 4). If* $\lim_{N\to\infty} \varepsilon_N = 0$ *and* $\lim_{N\to\infty} p_N = 1$*, we have that*

$$\lim_{N\to\infty} \mathbb{P}(\ell_N > (1 + \Delta)\ell(x^*)) = 0$$

for any $\Delta > 0$ *and that*

$$\lim_{N\to\infty} \mathbb{P}(|\mathcal{S}_N| < |\mathcal{I}^*|) = 0.$$

7 Results

We evaluated IRIS on three simulated scenarios: (1) a planar manipulator inspecting the boundary of a square region (Fig. 7a), (2) an unmanned aerial vehicle (UAV) inspecting the outer surface of a bridge (Fig. 7b), and (3) a CRISP robot inspecting the inner surface of a pleural cavity (Fig. 7c). For all experiments, we order path pairs in OPEN (Alg. 1 line 4) according to the path pair with the minimal potentially achievable path cost. All tests were run on a 3.4GHz 8-core Intel Xeon E5-1680 CPU with 64GB of RAM.

7.1 Planar manipulator scenario

In this scenario, depicted in Fig. 7a, we have a 5-link planar manipulator fixed at its base that is tasked with inspecting the boundary of a rectangular 2D workspace, which is discretized into 400 POI. The sensor is a camera attached to the tip of the manipulator, aligning with the direction of the robot's final segment. When modeling the camera for inspection, we consider a field of view (FOV) of 45 degrees and an unbounded effective inspecting distance. We start by evaluating IRIS for fixed p and ε and then compare it with RRTOT using our approach for dynamically reducing the approximation factors. For every set of parameters, we ran ten experiments for 1000 seconds and report the average value together with the standard deviation.

This scenario serves as a simple example where we compare the approximation algorithm for inspection planning with optimal inspection planning (i.e., $p = 1, \varepsilon = 0$). When $p = 1$, indicating we do not allow any approximation on inspection coverage, and we vary ε (Fig. 8a), we can see that even small approximation factors (e.g., $\varepsilon = 0.5$) allow to dramatically increase the coverage obtained as each search episode takes less time and more configurations can be added to the RRT tree. While optimal inspection planning (using $\varepsilon = 0$) did not result in 80% coverage even after 1000 seconds, this was achieved within one second for $\varepsilon \geq 1.0$. This comes at the price of slightly longer inspection paths. When $\varepsilon = 0$, indicating we do not allow any approximation on path length, and we vary p (Fig. 8b), we get roughly the same coverage per time but at the price of much longer paths for higher values of p.

Following the above discussion, when reaching high coverage is the sole objective, one should use large initial values of p_0 and ε_0. When we want initial solutions to also be short, one should start with smaller approximation factors. We compared IRIS with different initial approximation factors to RRTOT (Bircher et al. 2017), see Fig. 8c. We can see that our approach allows producing higher-quality paths than RRTOT. For example, IRIS obtains more than a $264\times$ speedup when compared to RRTOT when producing the same quality of inspection planning for the case of roughly 85% coverage and path length of 85.8 units. Final inspection paths obtained by IRIS are both shorter and inspect larger portions of $\mathcal{I}$.

7.2 Bridge inspection scenario

In this scenario, depicted in Fig. 7b, a UAV equipped with a camera inspects the surface of a bridge structure. The bridge structure is obtained from a 3D mesh (Elkassar 2008) and discretized into 3817 POI. The bridge structure serves as both an inspection target and an obstacle that may block movements and occlude sensing. And we do not consider other environmental obstacles except for the ground, which means the UAV can only fly above the ground.

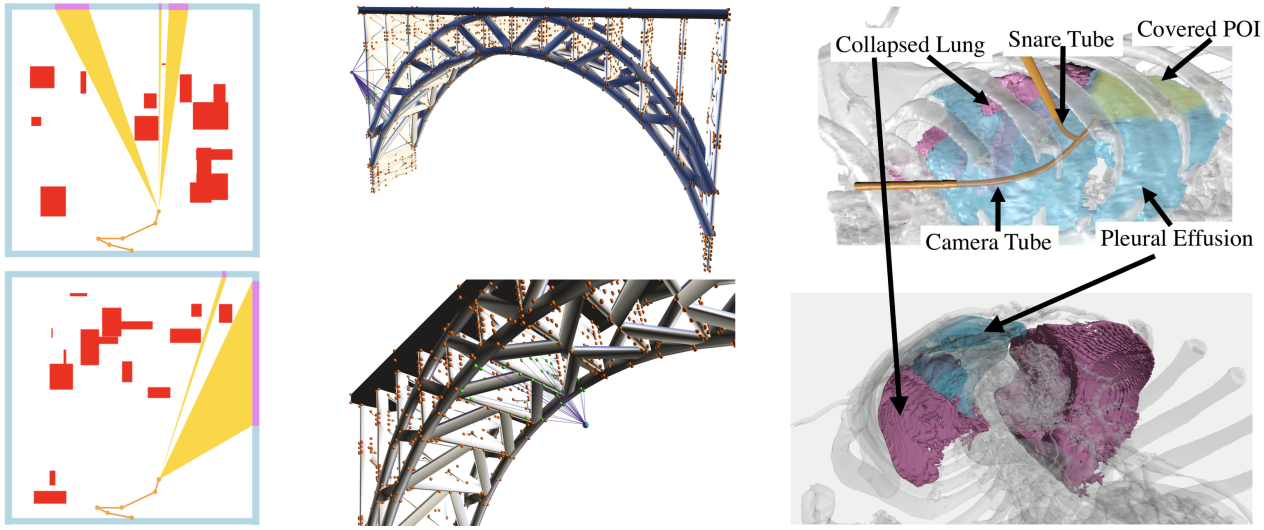
The UAV has a configuration space of $\mathcal{SO}(2) \times \mathbb{R}^3$. It first can translate in 3D space, then can rotate around its vertical axis, and finally, the camera can rotate around the pitch axis. When modeling the camera for inspection, we consider a FOV of 90 degrees and an effective inspecting distance of 10 meters.

We ran IRIS and RRTOT for this scenario ten different times for 10,000 seconds (Fig. 9). IRIS obtains more than a $8\times$ speedup when compared to RRTOT when producing a better quality of inspection planning for the case of roughly 57% coverage and path length of 230 units, which is only 68% of that of RRTOT.

7.3 Pleural effusion inspection scenario

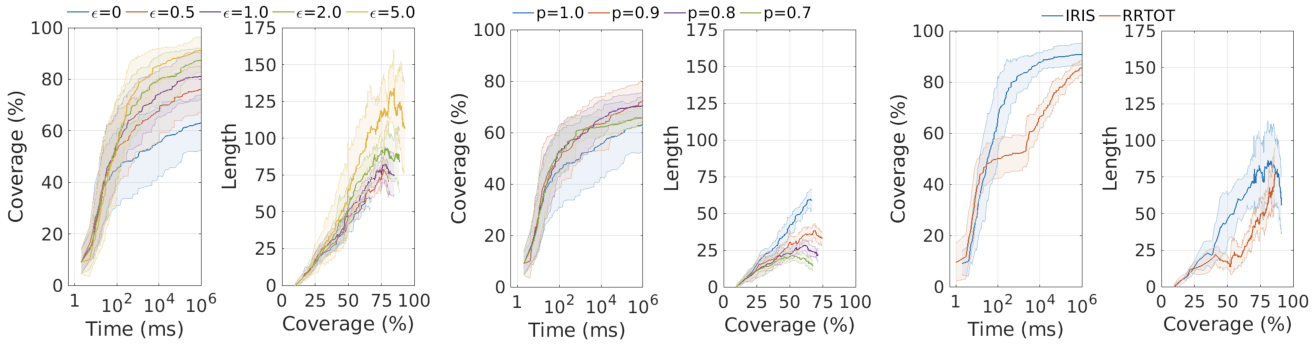
The anatomical pleural effusion environment for this simulation scenario was obtained from a Computed Tomography (CT) scan of a real patient suffering from this condition, and a fine discretization of the internal surface of the pleural cavity is used as the set of POI containing 49506 points. We also use the internal surface of the cavity as obstacles and prohibit the robot from colliding with the pleural surface, lung, and chest wall (except at tube entry points). Pleural effusion volumes can be geometrically complex, as the way in which the lung separates from the chest wall can be inconsistent. This results in unseparated regions of the lung's surface that can inhibit movement and occlude the sensor from visualizing areas further in the volume.

Here we consider a CRISP robot with two tubes, where a snare tube is grasping a camera tube in order to create a parallel structure made of thin, flexible tubes. Each tube can be independently rotated in three dimensions about its entry point into the body, and independently translated into and out of the cavity. The system has 8 degrees of freedom with a configuration space of $\mathcal{SO}(3)^2 \times \mathbb{R}^2$, which enables the



(a) Planar manipulator **(b)** Bridge inspection **(c)** Pleural effusion inspection

Figure 7. Simulation scenarios. (a) A 5-link planar manipulator (orange) inspects the boundary of a square region (blue) where rectangular obstacles (red) may block the robot and occlude the sensor. The sensor's field of view (FOV) is represented by the yellow region. $\mathcal{S}(\mathbf{q})$ are the points on the boundary in the sensor's unobstructed FOV and are shown in purple. (b) The bridge inspection scenario involves a UAV (blue) inspecting the outer surface of a bridge, including the POI that are covered (green) and non-covered (orange) from the current configuration. (c) The pleural effusion inspection scenario involves the CRISP robot (orange) inspecting the inner surface of a pleural cavity, including the POI that are covered (green) and non-covered (blue) from the current robot configuration.



(a) **(b)** **(c)**

Figure 8. Quality of inspection paths computed for the planar manipulator. The algorithm falls back to optimal inspection planning when $p = 1, \varepsilon = 0$. (a) IRIS running with $p = 1$, $f = 0$ and varying values of ε. (b) IRIS running with $\varepsilon = 0$, $f = 0$ and varying values of p. (c) Comparison of IRIS and RRTOT. IRIS running with $p_0 = 0.95$, $\varepsilon_0 = 20.0$, and $f = 0.0005$.

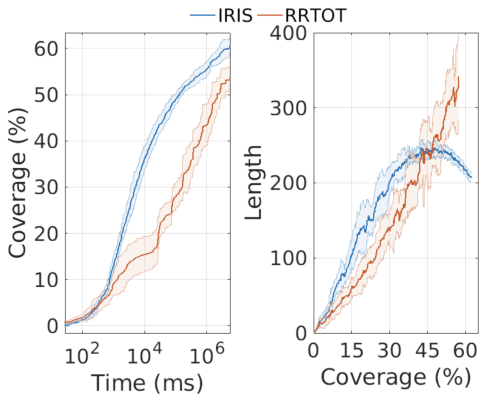


Figure 9. Comparing the quality of inspection paths computed for the bridge inspection scenario. IRIS was run with $p_0 = 0.7$, $\varepsilon_0 = 5$, and $f = 0.0001$.

parallel structure to move in a manner that enables obstacle avoidance as well as precise control of the camera's pose. We model the camera with a 60-degree FOV and an unbounded effective inspecting distance.

We ran IRIS and RRTOT for this scenario ten different times for 10,000 seconds (Fig. 10). Similar to the planar manipulator scenario, IRIS allows producing higher-quality paths than RRTOT. For example, IRIS obtains more than a $79\times$ speedup when compared to RRTOT when producing the same quality of inspection planning for the case of roughly 33% coverage and path length of 0.3 units.

8 Conclusion and Future Work

In this work, we presented IRIS, an algorithmic framework for computing asymptotically optimal inspection plans. Our key contribution is an algorithm to efficiently compute near-optimal inspection plans on graphs. Interestingly, our

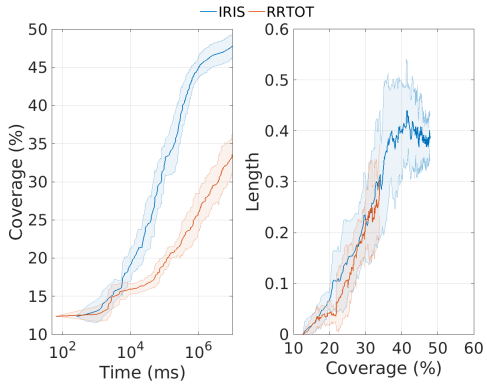


Figure 10. Comparing the quality of inspection paths computed for the pleural effusion scenario. IRIS was run with $p_0 = 0.8$, $\varepsilon_0 = 10$, and $f = 0.01$.

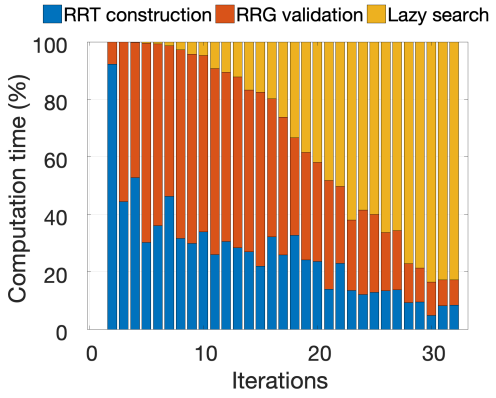


Figure 11. Time decomposition of IRIS as a function of iteration number.

problem of graph-inspection planning lies in the intersection between single and bi-criteria shortest path problems. Clearly, we are computing a shortest path on the inspection graph $\mathcal{G}_\mathcal{S}$. However, en route, we compute an approximation of the set of Pareto optimal paths to every node in the original graph $\mathcal{G}$. Thus, we believe that our approach may be useful for the general problem of bicriteria optimization.

We showed IRIS outperforms the prior state-of-the-art, including in a medical application in which a surgical robot inspects a tissue surface inside the body as part of a diagnostic procedure. However, the efficiency of IRIS can be further improved. We now highlight several avenues where such improvement could be obtained.

8.1 Dynamic updates in graph inspection planning

IRIS reruns Alg. 1 every iteration which may be highly inefficient as we would like to reuse information constructed from previous search episodes. Indeed, the general case where a graph undergoes a series of edge insertions and edge deletions and we wish to update a shortest-path algorithm is a well-studied problem referred to as the *fully dynamic single-source shortest-path problem* (Frigioni et al. 2000; Ramalingam and Reps 1996). Efficient algorithms exist even when running an A*-like search (Koenig et al. 2004). Thus, an immediate next step to improve the efficiency of our algorithm is to adapt the aforementioned algorithms to the case of near-optimal graph inspection planning.

8.2 Balancing graph search and lazy computation

Recall that we employ a lazy search paradigm when computing near-optimal inspection plans on the inspection graph (Sec. 5.4). This was done because edge evaluation is computationally complex. However, as the number of iterations increases, the search starts to dominate the overall running time of our algorithm and *not* edge evaluation (see Fig. 11). Recently Mandalika et al. (2018, 2019) presented an algorithm that balances edge evaluation and graph search when edges are expensive to evaluate using the notion of *lazy look-ahead*. Thus, we suggest using their method *dynamically* varying the so-called lazy look-ahead—in the initial stages of the algorithm, when the search is not a bottleneck, employ a large look-ahead (which corresponds to performing more search). As the algorithm progress, reduce the look-ahead to account for the fact that edge evaluation is relatively cheaper than graph search.

8.3 Efficient sampling of configurations in RRT construction

Recall that in our RRT constructions we sample configurations uniformly at random from $\mathcal{X}$. Common implementations of RRT typically employ a *goal bias* where configurations from the goal are sampled with some probability LaValle (2006). Similarly, we suggest biasing sampling towards configurations that increase coverage. Namely, to configurations $\mathbf{q}$ such that $\mathcal{S}(\mathbf{q}) \cup \mathcal{I}_G \neq \emptyset$. We suspect that the goal bias should be dynamically changed—when the inspection graph $\mathcal{G}_\mathcal{S}$ has low coverage the bias should be high. As the overall coverage of $\mathcal{G}_\mathcal{S}$ increases, the goal bias should be reduced to allow for shorter inspection plans.

8.4 Employing multiple heuristics in graph inspection planning

As the number of iterations increases, graph search dominates the running time of our algorithm. Heuristics have been shown to be an effective tool in speeding up search algorithms and we suggest employing recent developments from the search community to speed up this part of our framework. One such development is using *multiple* heuristics to guide the search in a systematic way (Aine et al. 2016) that has shown to be an effective tool in robot planning algorithms (Islam et al. 2018; Ranganeni et al. 2018).

Roughly speaking, using multiple heuristics allows encoding domain knowledge without having to worry about the heuristic functions being admissible. In our setting, we are simultaneously reasoning about inspection coverage and plan length in our graph inspection planning. Thus, it may be beneficial to design one (or more) heuristics that account for path length and one (or more) heuristics that account for path coverage. Then we could apply a method similar to MHA* (Aine et al. 2016) to combine the efforts of these heuristics.

8.5 Adaptively updating approximation parameters

In our work, we used a simplistic approach to update the approximation parameters. These may have a dramatic effect on the quality of plans produced. We suggest further inspecting how to update these parameters, possibly doing this in a dynamic fashion according to information obtained from previous search episodes.

Acknowledgements

The authors gratefully acknowledge Inbar Fried at the University of North Carolina at Chapel Hill for his comments regarding the proofs.

Declaration of conflicting interests

The authors declare that there is no conflict of interest.

Funding

The authors disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research was supported in part by the National Science Foundation (NSF) [grant numbers 2008475, 2038855]; the Israeli Ministry of Science, Technology and Space (MOST) [grant numbers 3-16079, 3-17385]; and the United States-Israel Binational Science Foundation (BSF) [grant numbers 2019703, 2021643].

References

Aine S, Swaminathan S, Narayanan V, Hwang V and Likhachev M (2016) Multi-heuristic A*. *Int. J. Robotics Research (IJRR)* 35(1-3): 224–243.

Almadhoun R, Taha T, Seneviratne L, Dias J and Cai G (2016) A survey on inspecting structures using robotic systems. *Int. J. Advanced Robotic Systems* 13(6).

Anderson PL, Mahoney AW and Webster III RJ (2017) Continuum reconfigurable parallel robots for surgery: Shape sensing and state estimation with uncertainty. *IEEE Robotics and Automation Letters* 2(3): 1617–1624.

ASCE (2016) ASCE 2017 infrastructure report card. `https://www.infrastructurereportcard.org/wp-content/uploads/2016/10/2017-Infrastructure-Report-Card.pdf`. Accessed: 2019-01-01.

Bingham B, Foley B, Singh H, Camilli R, Delaporta K, Eustice R, Mallios A, Mindell D, Roman C and Sakellariou D (2010) Robotic tools for deep water archaeology: Surveying an ancient shipwreck with an autonomous underwater vehicle. *J. Field Robotics* 27(6): 702–717.

Bircher A, Alexis K, Burri M, Oettershagen P, Omari S, Mantel T and Siegwart R (2015) Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, pp. 6423–6430.

Bircher A, Alexis K, Schwesinger U, Omari S, Burri M and Siegwart R (2017) An incremental sampling–based approach to inspection planning: The rapidly–exploring random tree of trees. *Robotica* 35(6): 1327–1340.

Bircher A, Kamel M, Alexis K, Burri M, Oettershagen P, Omari S, Mantel T and Siegwart R (2016) Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots. *Autonomous Robots* 40(6): 1059–1078.

Bogaerts B, Sels S, Vanlanduit S and Penne R (2018) A gradient-based inspection path optimization approach. *IEEE Robotics and Automation Letters* 3(3): 2646–2653.

Bohlin R and Kavraki LE (2000) Path planning using lazy PRM. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*. pp. 521–528.

Chen P and Nie Y (2013) Bicriterion shortest path problem with a general nonadditive cost. *Transportation Research Part B: Methodological* 57: 419–435.

Choset HM, Hutchinson S, Lynch KM, Kantor G, Burgard W, Kavraki LE and Thrun S (2005) *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT press.

Danner T and Kavraki LE (2000) Randomized planning for short inspection paths. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*. pp. 971–976.

Dellin CM and Srinivasa SS (2016) A unifying formalism for shortest path problems with expensive edge evaluations via lazy best-first search over paths with edge selectors. In: *Int. Conf. Automated Planning and Scheduling (ICAPS)*. pp. 459–467.

Dijkstra EW (1959) A note on two problems in connexion with graphs. *Numerische Mathematik* 1(1): 269–271.

Edelkamp S, Pomarlan M and Plaku E (2017) Multiregion inspection by combining clustered traveling salesman tours with sampling-based motion planning. *IEEE Robotics and Automation Letters* 2(2): 428–435.

Ehrgott M and Gandibleux X (2000) A survey and annotated bibliography of multiobjective combinatorial optimization. *OR-Spektrum* 22(4): 425–460.

Elkassar (2008) TurboSquid. `https://www.turbosquid.com/3d-models/bridge-max-free/424181`. Accessed: 2019-10-01.

Englot B and Hover FS (2012) Sampling-based coverage path planning for inspection of complex structures. In: *Int. Conf. Automated Planning and Scheduling (ICAPS)*. pp. 29–37.

Englot BJ and Hover FS (2011) Planning complex inspection tasks using redundant roadmaps. In: *Int. Symp. Robotics Research (ISRR)*. pp. 327–343.

Frigioni D, Marchetti-Spaccamela A and Nanni U (2000) Fully dynamic algorithms for maintaining shortest paths trees. *J. Algorithms* 34(2): 251–281.

Fu M, Kuntz A, Salzman O and Alterovitz R (2019) Toward asymptotically-optimal inspection planning via efficient near-optimal graph search. In: *Robotics: Science and Systems (RSS)*. FreiburgimBreisgau, Germany. DOI:10.15607/RSS.2019.XV.057.

Fu M, Salzman O and Alterovitz R (2021) Computationally-efficient roadmap-based inspection planning via incremental lazy search. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, pp. 7449–7456.

Galceran E and Carreras M (2013) A survey on coverage path planning for robotics. *Robotics and Autonomous Systems* 61(12): 1258–1276.

Gracias N, Ridao P, Garcia R, Escartín J, L'Hour M, Cibecchini F, Campos R, Carreras M, Ribas D, Palomeras N, Magi L, Palomer A, Nicosevici T, Prados R, Hegedüs R, Neumann L, de Filippo F and Mallios A (2013) Mapping the moon: Using a lightweight auv to survey the site of the 17th century ship 'la

lune'. In: *OCEANS-Bergen, 2013 MTS/IEEE*. IEEE, pp. 1–8.

Haghtalab N, Mackenzie S, Procaccia AD, Salzman O and Srinivasa SS (2018) The provable virtue of laziness in motion planning. In: *Int. Conf. Automated Planning and Scheduling (ICAPS)*. pp. 106–113.

Halperin D, Salzman O and Sharir M (2018) Algorithmic motion planning. In: *Handbook of Discrete and Computational Geometry, Third Edition.* CRC Press LLC, pp. 1311–1342.

Hart PE, Nilsson NJ and Raphael B (1968) A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Systems, Science, and Cybernetics* 4(2): 100–107.

Hauser K (2015) Lazy collision checking in asymptotically-optimal motion planning. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*. pp. 2951–2957.

Hernández C, Yeoh W, Baier JA, Zhang H, Suazo L, Koenig S and Salzman O (2023) Simple and efficient bi-objective search algorithms via fast dominance checks. *Artificial Intelligence* 314: 103807.

Islam F, Salzman O and Likhachev M (2018) Online, interactive user guidance for high-dimensional, constrained motion planning. In: *Int. Joint Conf. on Artificial Intelligence (IJCAI)*. pp. 4921–4928.

Johnson-Roberson M, Pizarro O, Williams SB and Mahon I (2010) Generation and visualization of large-scale three-dimensional reconstructions from underwater robotic surveys. *J. Field Robotics* 27(1): 21–51.

Kafka P, Faigl J and Váňa P (2016) Random inspection tree algorithm in visual inspection with a realistic sensing model and differential constraints. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*. pp. 2782–2787.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *Int. J. Robotics Research (IJRR)* 30(7): 846–894.

Kavraki LE, Svestka P, Latombe JC and Overmars MH (1996) Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Trans. Robotics and Automation* 12(4): 566–580.

Koenig S, Likhachev M and Furcy D (2004) Lifelong planning A*. *Artificial Intelligence* 155(1-2): 93–146.

Kuffner JJ and LaValle SM (2000) RRT-connect: An efficient approach to single-query path planning. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*, volume 2. pp. 995–1001.

Kuntz A, Bowen C, Baykal C, Mahoney AW, Anderson PL, Maldonado F, Webster III RJ and Alterovitz R (2018) Kinematic design optimization of a parallel surgical robot to maximize anatomical visibility via motion planning. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*. pp. 926–933.

Latombe JC (1991) *Robot Motion Planning*. Boston, MA: Kluwer.

LaValle SM (2006) *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press.

LaValle SM and Kuffner JJ (2001) Randomized kinodynamic planning. *Int. J. Robotics Research (IJRR)* 20(5): 378–400.

Li Y, Littlefield Z and Bekris KE (2016) Asymptotically optimal sampling-based kinodynamic planning. *Int. J. Robotics Research (IJRR)* 35(5): 528–564.

Likhachev M, Gordon GJ and Thrun S (2003) Ara*: Anytime a* with provable bounds on sub-optimality. *Advances in Neural Information Processing Systems* 16.

Lynch KM and Park FC (2017) *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press.

Mahoney AW, Anderson PL, Swaney PJ, Maldonado F and Webster III RJ (2016) Reconfigurable parallel continuum robots for incisionless surgery. In: *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*. pp. 4330–4336.

Mandalika A, Choudhury S, Salzman O and Srinivasa S (2019) Generalized lazy search for robot motion planning: Interleaving search and edge evaluation via event-based toggles. In: *Int. Conf. Automated Planning and Scheduling (ICAPS)*, volume 29. pp. 745–753.

Mandalika A, Salzman O and Srinivasa S (2018) Lazy receding horizon A* for efficient path planning in graphs with expensive-to-evaluate edges. In: *Int. Conf. Automated Planning and Scheduling (ICAPS)*. pp. 476–484.

Marble JD and Bekris KE (2011) Asymptotically near-optimal is good enough for motion planning. In: *Int. Symp. Robotics Research (ISRR)*. pp. 419–436.

Niyaz S, Kuntz A, Salzman O, Alterovitz R and Srinivasa S (2018) Following surgical trajectories with concentric tube robots via nearest-neighbor graphs. In: *Int. Symp. Experimental Robotics (ISER)*.

Papadopoulos G, Kurniawati H and Patrikalakis NM (2013) Asymptotically optimal inspection planning using systems with differential constraints. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, pp. 4126–4133.

Pardalos PM, Migdalas A and Pitsoulis L (2008) *Pareto optimality, game theory and equilibria*, volume 17. Springer Science & Business Media.

Pohl I (1970) Heuristic search viewed as path finding in a graph. *Artificial Intelligence* 1(3-4): 193–204.

Raffaeli R, Mengoni M, Germani M and Mandorli F (2013) Off-line view planning for the inspection of mechanical parts. *Int. J. Interactive Design and Manufacturing (IJIDeM)* 7(1): 1–12.

Ramalingam G and Reps T (1996) On the computational complexity of dynamic graph problems. *Theoretical Computer Science* 158(1&2): 233–277.

Ranganeni V, Salzman O and Likhachev M (2018) Effective footstep planning for humanoids using homotopy-class guidance. In: *Int. Conf. Automated Planning and Scheduling (ICAPS)*. pp. 500–508.

Reinhardt LB and Pisinger D (2011) Multi-objective and multi-constrained non-additive shortest path problems. *Computers & OR* 38(3): 605–616.

Richter S, Thayer JT and Ruml W (2010) The joy of forgetting: Faster anytime search via restarting. In: *Int. Conf. Automated Planning and Scheduling (ICAPS)*.

Salzman O and Halperin D (2015) Asymptotically-optimal motion planning using lower bounds on cost. In: *IEEE Int. Conf. Robotics and Automation (ICRA)*. pp. 4167–4172.

Salzman O and Halperin D (2016) Asymptotically near-optimal RRT for fast, high-quality motion planning. *IEEE Trans. Robotics* 32(3): 473–483.

Salzman O, Hou B and Srinivasa S (2017) Efficient motion planning for problems lacking optimal substructure. In: *Int. Conf. Automated Planning and Scheduling (ICAPS)*. pp. 531–539.

Schmerling E, Janson L and Pavone M (2015) Optimal sampling-based motion planning under differential constraints: The driftless case. In: *IEEE Int. Conf. Robotics and Automation*

(ICRA). pp. 2368–2375.

Solovey K, Salzman O and Halperin D (2018) New perspective on sampling-based motion planning via random geometric graphs. *Int. J. Robotics Research (IJRR)* 37(10): 1117 – 1133.

Tivey MA, Bradley A, Yoerger D, Catanach R, Duester A, Liberatore S and Singh H (1997) Autonomous underwater vehicle maps seafloor. *Eos, Tran. American Geophysical Union* 78(22): 229–230.

Tsaggouris G and Zaroliagis CD (2004) Non-additive shortest paths. In: *European Symposium on Algorithms (ESA)*. pp. 822–834.

Tsaggouris G and Zaroliagis CD (2009) Multiobjective optimization: Improved FPTAS for shortest paths and non-linear objectives with applications. *Theory of Computing Systems* 45(1): 162–186.

van den Berg J, Shah R, Huang A and Goldberg K (2011) Anytime nonparametric a*. In: *AAAI Conf. Artificial Intelligence*.

Zhang H, Salzman O, Kumar TKS, Felner A, Ulloa CH and Koenig S (2022) Anytime approximate bi-objective search. In: *Int. Symp. Combinatorial Search (SOCS)*. pp. 199–207.

Appendix A Near-optimal inspection graph search

Definition 1. Optimal inspection path on a roadmap. *Let $(\mathcal{G}, \mathcal{I}_\mathcal{G}, \mathcal{S}, \ell, v_\mathrm{s})$ be a graph inspection problem. An* optimal inspection path *P^* is a path on roadmap $\mathcal{G}$, starting at v_s, and satisfies*

$$\ell(P^*) = \operatorname{argmin}\{\ell(P) | P \text{ is a path with } \mathcal{S}(P) = \mathcal{I}_\mathcal{G}\}.$$

Denote the length and coverage of an optimal path as $\ell^* := \ell(P^*)$ and $\mathcal{S}^* := \mathcal{S}(P^*) = \mathcal{I}_\mathcal{G}$, respectively.

Lemma 1. *In near-optimal inspection graph search (Alg. 1), all path pairs in OPEN and CLOSE during the search are ε, p-bounded.*

Proof. Recall that the first path pair (line 2) is $\mathsf{PP}_s = (P_\mathrm{s}, P_\mathrm{s})$ (here P_s is a path containing only start vertex v_s). Namely, the potentially achievable path is identical to the achievable path of PP_s and the path pair is trivially ε, p-bounded.

We now show that the set of ε, p-bounded path pairs are closed under extending and subsuming operations proving the Lemma.

Closure under subsuming operation (line 18-21, 25-27): This holds trivially since subsuming only occurs after checking that the resultant path pair will be ε, p-bounded (line 18 and 25).

Closure under extending operation (line 9): Let PP_u be an ε, p-bounded path pair extended by edge $e = (u, v)$ to obtain the path pair PP_v (line 9). Let us consider both the length and coverage of PP_v.

By definition of the extending operation we have that (i) $\ell(P_v) = \ell(P_u) + \ell(e)$ and (ii) $\ell(\tilde{P}_v) = \ell(\tilde{P}_u) + \ell(e)$. Thus:

$$\begin{aligned}\ell(P_v) &= \ell(P_u) + \ell(e)\\ &\leq (1+\varepsilon)\cdot \ell(\tilde{P}_u) + \ell(e)\\ &\leq (1+\varepsilon)\cdot \Big(\ell(\tilde{P}_u) + \ell(e)\Big)\\ &= (1+\varepsilon)\cdot \ell(\tilde{P}_v)\end{aligned}$$

Here, the first and last equalities follow from the definition of the extend operator, the first inequality follows from the fact that PP_u is ε, p-bounded.

Moving to the coverage of the resultant path pair, by definition of the extend operator we have that (i) $\mathcal{S}(P_v) = \mathcal{S}(P_u) \cup \mathcal{S}(v)$ and (ii) $\mathcal{S}(\tilde{P}_v) = \mathcal{S}(\tilde{P}_u) \cup \mathcal{S}(v)$. Thus:

$$\begin{aligned}|\mathcal{S}(P_v)| &= |\mathcal{S}(P_u) \cup \mathcal{S}(v)|\\ &= |\mathcal{S}(P_u)| + |\mathcal{S}(v) \cap \overline{\mathcal{S}(P_u)}|\\ &\geq |\mathcal{S}(P_u)| + |\mathcal{S}(v) \cap \overline{\mathcal{S}(P_u)} \cap \overline{\mathcal{S}(\tilde{P}_u)}|\\ &\geq p\cdot|\mathcal{S}(P_u) \cup \mathcal{S}(\tilde{P}_u)| + |\mathcal{S}(v) \cap \overline{\mathcal{S}(P_u)} \cap \overline{\mathcal{S}(\tilde{P}_u)}|\\ &\geq p\cdot\Big(|\mathcal{S}(P_u) \cup \mathcal{S}(\tilde{P}_u)| + |\mathcal{S}(v) \cap \overline{\mathcal{S}(P_u)} \cap \overline{\mathcal{S}(\tilde{P}_u)}|\Big)\\ &= p\cdot|\mathcal{S}(P_v) \cup \mathcal{S}(\tilde{P}_v)|\end{aligned}$$

Here $\overline{\mathcal{S}(P)} := \mathcal{I}_\mathcal{G} \setminus \mathcal{S}(P)$, the first and last equalities follow from the definition of the extend operation, the second inequality follows from the fact that PP_u is ε, p-bounded.

Thus, an ε, p-bounded PP will remain ε, p-bounded after the extending operation. ■

Lemma 2. *Let $P^* = \{v_0, v_1, \ldots, v_{n^*}\}$ be an optimal inspection path.*[‡] *Denote $P^*[i] := \{v_0, \ldots, v_i\}$, for $i \in [0, n^*]$ as the path composed of the first i waypoints of P^*. During every iteration of Alg 1, there exists a path pair $\mathsf{PP}_v = (P_v, \tilde{P}_v)$ in the open list and an index i such that $v = v_i$ and $\tilde{P}_v$* strictly *dominates $P^*[i]$. Namely, $\ell(\tilde{P}) \leq \ell(P^*[i])$ and $\mathcal{S}(P^*[i]) \subseteq \mathcal{S}(\tilde{P})$.*

Proof. We will prove a slightly stronger claim showing that at every iteration of Alg 1 there exists a path pair $\mathsf{PP}_v = (P_v, \tilde{P}_v)$ in the open list such that the following two properties hold:

P1 There exists an index i such that $v = v_i$ and $\tilde{P}_v$ *strictly* dominates $P^*[i]$.

P2 There is no index $j > i$ such that there is another path pair $\mathsf{PP}_u = (P_u, \tilde{P}_u)$ in either the open or the closed list where $u = v_j$ and $\tilde{P}_u$ *strictly* dominates $P^*[j]$.

Property **P1** is exactly what we need to prove for Lemma 2 while property **P2** simplifies the proof. The proof will be by induction on the iterations of Alg. 1.

Induction base: In line 2, the open set is initialized with the path pair $\mathsf{PP}_s = (P_s, P_s)$. P_s trivially dominates $P^*[0]$ (property **P1**). Furthermore, there is no path in the closed list so property **P2** holds trivially.

Induction step: Assume that at iteration i, there exists some index $n_i \in [0, n^*]$ and some path pair $\mathsf{PP}_{n_i} = (P_{n_i}, \tilde{P}_{n_i})$ such that $\tilde{P}_{n_i}$ dominates $P^*[n_i]$. Note that following property **P2**, there is no other path pair $\mathsf{PP}_u = (P_u, \tilde{P}_u)$ in either the open or the closed list and an index $n_j > n_i$ such that $u = v_{n_j}$ and $\tilde{P}_u$ *strictly* dominates $P^*[n_j]$. There are two cases to consider—either PP_{n_i} was popped from the open list or not.

C1 First, we consider the case where PP_{n_i} was popped from the open list. Consider the set $\mathcal{V}_\mathrm{nbr}$ of neighbors

[‡]There may be more than one optimal path and it is possible that $n^* \geq |\mathcal{V}|$ since P^* may revisit vertices on $\mathcal{G}$.

of v_{n_i} that lie on P^*. Namely,

$$\mathcal{V}_{\text{nbr}} = \{v_{n_i+k} | (v_{n_i}, v_{n_i+k}) \in \mathcal{E} \text{ and } k \geq 1\}.$$

Note that $\mathcal{V}_{\text{nbr}}$ is not empty as $e_{i+1} = (v_{n_i}, v_{n_i+1}) \in \mathcal{E}$. For each vertex $v_{n_i+k} \in \mathcal{V}_{\text{nbr}}$, denote $\mathsf{PP}_{n_i+k} = (P_{n_i+k}, \tilde{P}_{n_i+k})$ the path pair that will be generated after extending the edge (v_{n_i}, v_{n_i+k}) (line 9). Let k^* be the maximal index such that (i) $v_{n_i+k^*} \in \mathcal{V}_{\text{nbr}}$ and (ii) $\tilde{P}_{n_i+k^*}$ strictly dominates $P^*[n_i + k^*]$. Note that k^* exists since $\tilde{P}_{n_i+1}$ strictly dominates $P^*[n_i + 1]$ (this is easily verified using the induction hypothesis). Assume that $\mathsf{PP}_{n_i+k^*}$ was the first path pair that was generated[§] (line 8) and that the induction hypothesis holds prior to this event. It is straightforward to see that both properties hold for $\mathsf{PP}_{n_i+k^*}$. Now, we need to consider the following sub-cases:

C1.1 $P_{n_i+k^*}$ is *strictly* dominated by some path $P'_{n_i+k^*}$ that is part of the path pair $\mathsf{PP}'_{n_i+k^*} = (P'_{n_i+k^*}, \tilde{P}'_{n_i+k^*})$ in the closed list (line 12). Note that $\tilde{P}'_{n_i+k^*}$ strictly dominates $P^*[n_i + k^*]$ and recall that $\tilde{P}_{n_i+k^*}$ strictly dominates $P^*[n_i + k^*]$. Thus, $\tilde{P}'_{n_i+k^*}$ strictly dominates $P^*[n_i + k^*]$ in contradiction to property **P2** meaning that this cannot occur.

C1.2 $\mathsf{PP}_{n_i+k^*}$ is subsumed by some other path pair $\mathsf{PP}_{\text{open}} = (P_{\text{open}}, \tilde{P}_{\text{open}})$ in the open list (lines 17-21). The resulting path pair PP_{res} from this subsuming operation has $\ell(\tilde{P}_{\text{res}}) = \min\{\ell(\tilde{P}_{n_i+k^*}), \ell(\tilde{P}_{\text{open}})\} \leq \ell(\tilde{P}_{n_i+k^*})$ and $\mathcal{S}(\tilde{P}_{\text{res}}) = \mathcal{S}(\tilde{P}_{n_i+k^*}) \cup \mathcal{S}(\tilde{P}_{\text{open}}) \supseteq \mathcal{S}(\tilde{P}_{n_i+k^*})$. Thus, as $\tilde{P}_{\text{res}}$ strictly dominates $\tilde{P}_{n_i+k^*}$ it also strictly dominates $P^*[n_i + k^*]$. This, in turn, implies that both properties hold.

C1.3 $\mathsf{PP}_{n_i+k^*}$ is subsuming some other path in the open list (lines 24-27). Similar to **C1.2**, the resulting path pair from subsuming operation has potentially achievable path that strictly dominates $\tilde{P}_{n_i+k^*}$, thus strictly dominates $P^*[n_i + k^*]$. Again, this implies that both properties hold.

C1.4 $\mathsf{PP}_{n_i+k^*}$ is inserted into the open list without being subsumed or subsuming any other path pair. Here the induction hypothesis trivially holds.

We now need to consider all other path pairs generated throughout. However, non can result in a path pair that subsumes $\mathsf{PP}_{n_i+k^*}$ or is subsumed by $\mathsf{PP}_{n_i+k^*}$ meaning that the induction hypothesis still holds.

C2 Now consider the case that PP_{n_i} was *not* popped from the priority queue. Some other path pair was popped from the priority and extended and pushed into the closed list. Out of all newly created path pairs (line 9) let PP_{n_j} be the one for which property **P1** holds and for which the index n_j is maximal. If no such path pair exists then the induction hypothesis continues to hold. Again, we will consider several sub-cases.

C2.1 If $n_j < n_i$, then both property **P1** and **P2** still hold for PP_{n_i}.

C2.2 If $n_j = n_i$[¶], then either (i) PP_{n_i} was subsumed by PP_{n_j}, (ii) PP_{n_j} was subsumed by PP_{n_i} or (iii) no path pair was subsumed. It is straightforward to see that in all cases both property **P1** and **P2** still hold for the newly created path pair in cases (i) and (ii) or for both path pairs PP_{n_i} and PP_{n_j} on case (iii).

C2.3 If $n_j > n_i$, then both property **P1** and **P2** hold for PP_{n_j}.

Similar to **C1**, we need to consider all other path pairs generated throughout. However, non can result in a path pair that subsumes PP_{n_j} or is subsumed by PP_{n_j} meaning that the induction hypothesis still holds.

■

Note: Lemma 2 and its proof were stated for the optimal inspection path P^*. However, the proof holds for any path P.

We can order path pairs in the open list (line 4) either according to (i) the path pair with the minimal achievable path cost or (ii) the path pair with the minimal potentially achievable path cost. We now show that if either method is used, the path returned by the algorithm (line 7) ε, p-dominates an optimal inspection path P^*.

Theorem 1. Near-optimal inspection graph search. *Near-optimal inspection graph search (Alg. 1) computes a path P that ε, p-dominates an optimal inspection path P^*. Namely, $\ell(P) \leq (1 + \varepsilon) \cdot \ell^*$ and $|\mathcal{S}(P)| \geq p \cdot |\mathcal{S}^*|$.*

Proof. When Alg. 1 terminates (line 6-7), we have that $\mathcal{S}(\tilde{P}_u) = \mathcal{I}_{\mathcal{G}} = \mathcal{S}^*$. According to Lemma 1, PP_u is ε, p-bounded, thus $\ell(P_u) \leq (1 + \varepsilon) \cdot \ell(\tilde{P}_u)$ and $|\mathcal{S}(P_u)| \geq p \cdot |\mathcal{S}(P_u) \cup \mathcal{S}(\tilde{P}_u)| = p \cdot |\mathcal{S}^*|$.

According to Lemma 2, for an optimal inspection path P^* there always exists a path pair in the open set and an index i such that $\tilde{P}$ dominates $P^*[i]$. If the terminating $\mathsf{PP}_u = (P_u, \tilde{P}_u)$ (on line 6-7) happens to be such a path pair, it is straightforward that $\ell(P_u) \leq (1 + \varepsilon) \cdot \ell(\tilde{P}_u) \leq (1 + \varepsilon) \cdot \ell(P^*[i]) \leq (1 + \varepsilon) \cdot \ell^*$. Otherwise, such a path pair $\mathsf{PP}' = (P', \tilde{P}')$ is still in the open set.

When using the achievable cost to order the open list, we have that

$$\begin{aligned} \ell(P_u) &\leq \ell(P') \\ &\leq (1 + \varepsilon) \cdot \ell(\tilde{P}') \\ &\leq (1 + \varepsilon) \cdot \ell(P^*[i]) \\ &\leq (1 + \varepsilon) \cdot \ell^* \end{aligned}$$

[§]The assumption that $\mathsf{PP}_{n_i+k^*}$ was the first path pair that was generated is not required for Lemma 2 to hold but it simplifies the proof.

[¶]Strictly speaking, when $n_j = n_i$, the notation PP_{n_i} and PP_{n_j} is ambiguous. However, to simplify notation, we continue to refer to PP_{n_i} as the path pair for which the induction hypothesis holds and to PP_{n_j} as the newly created path pair.

When using potentially achievable costs to order the open list, we have

$$\begin{aligned}\ell(P_u) &\leq (1+\varepsilon)\cdot\ell(\tilde{P}_u)\\ &\leq (1+\varepsilon)\cdot\ell(\tilde{P}')\\ &\leq (1+\varepsilon)\cdot\ell(P^*[i])\\ &\leq (1+\varepsilon)\cdot\ell^*\end{aligned}$$

■

Appendix B Asymptotically optimal inspection planning

To prove that IRIS is asymptotically optimal, we will show that the optimal inspection path x^* can be approximated by a sequence of configurations sampled by our algorithm. Here, we will need to show that as the number of iterations approaches infinity, the following requirements hold: (i) the length of the path induced by this sequence of samples converges asymptotically to the length of x^*, (ii) the coverage obtained by this sequence of samples converges asymptotically to the coverage of x^* and that (iii) our graph-inspection algorithm finds such a sequence of samples.

To do so, we rely on the notion of *probabilistic exhaustivity* (Schmerling et al. 2015). Roughly speaking, it is the notion that given a sufficiently large set of uniformly sampled configurations, any path can be traced arbitrarily well in the configuration space by a path defined as a sequence of configurations. The original definition of tracing (to be formalized shortly) was used in the context of path cost and, as we will see, is insufficient for our purposes. Thus, we start in Sec. A in extending the definition of tracing and showing that probabilistic exhaustivity still holds for the extended version. This lays the groundwork to show that (i) and (ii) hold.

We then continue in Sec. B to formally define the well-behaving of an inspection trajectory. Together with the notion of tracing, well-behaving lays the groundwork to show that (ii) holds. Roughly speaking, this will ensure that there are no singular points along a trajectory where a POI can only be inspected from. This, in turn, will allow us to ensure that trajectories that trace an optimal inspection path cover the same set of POI.

Finally, in Sec. C, we come to the conclusion that IRIS is asymptotically optimal.

A Probabilistic exhaustivity

In order for our proof to be applicable to general systems with differential constraints, we need to introduce several notations. These will be used to prove the notion of probabilistic exhaustivity (Schmerling et al. 2015) for our setting. We start by defining (following (Schmerling et al. 2015, Sec. IV)) the *arc length* of a path and *sub-Riemmanian distances*.

Definition 2. Trajectory arc length. *Let $x:[0,T]\to\mathcal{X}$ be a continuous trajectory, the arc length of x is defined as*

$$\ell(x) := \int_0^T \|\dot{x}(t)\|dt.$$

Here, $\|\dot{x}(t)\| = \sqrt{\langle\dot{x}(t),\dot{x}(t)\rangle}$ is the squared root of the standard Euclidean inner product. See Fig. 1.

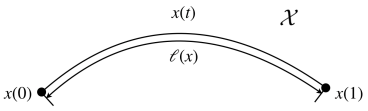


Figure 1. Arc length of a trajectory.

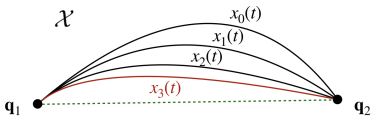


Figure 2. Distances between two configurations. Here $x_0(t),\ldots,x_3(t)$ are all dynamically feasible trajectories, and $x_3(t)$ is a trajectory with the shortest arc length, among all dynamically feasible trajectories connecting $\mathbf{q}_1$ and $\mathbf{q}_2$. So the sub-Riemmanian distance $d_{sr}(\mathbf{q}_1,\mathbf{q}_2)$ is the arc length of $x_3(t)$. The Euclidean distance $\|\mathbf{q}_1-\mathbf{q}_2\|$ is the length of the green dotted line.

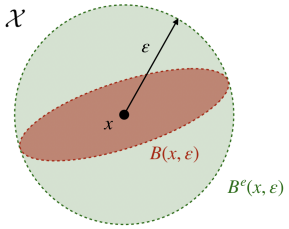


Figure 3. One *possible* example showing a sub-Riemannian ball and a Euclidean ball with the same center and radius.

Definition 3. Sub-Riemmanian distance. *Let $\mathbf{q}_1,\mathbf{q}_2\in\mathcal{X}$ be two configurations, the sub-Riemmanian distance between them is defined as*

$$d_{sr}(\mathbf{q}_1,\mathbf{q}_2) := \inf_x \ell(x).$$

Namely, it is the length of the shortest dynamically feasible trajectory[‖] $x(t)$ connecting $\mathbf{q}_1$ and $\mathbf{q}_2$. See Fig. 2.

Given the above definitions, we define the *Euclidean ball* on $\mathcal{X}$ as

$$B^e(x,\varepsilon) = \{y\in\mathcal{X} : \|x-y\|\leq\varepsilon\},$$

and the *sub-Riemannian ball* on $\mathcal{X}$ as

$$B(x,\varepsilon) = \{y\in\mathcal{X} : d_{sr}(x,y)\leq\varepsilon\}.$$

Note that by definition it holds that $\forall x,y\ \ \|x-y\|\leq d_{sr}(x,y)$ and $\forall x,\varepsilon\ B(x,\varepsilon)\subseteq B^e(x,\varepsilon)$. See Fig. 3.

The result of our algorithm, as is common in sampling-based methods, is a trajectory that is implicitly defined by a discrete sequence of configurations (a.k.a. waypoints). The following definition formalizes the continuous path associated with a discrete sequence of waypoints.

Definition 4. Associated optimal trajectory. *Let $P=\{\mathbf{q}_i\}_{i=1}^n\subseteq\mathcal{X}$ be a discrete sequence of configurations. The*

[‖] Roughly speaking, a trajectory is said to be dynamically feasible if there is a control function that satisfies the kinematic constraints of the system. For a precise definition, see (Schmerling et al. 2015, Sec. II).

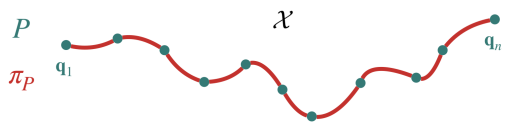


Figure 4. The associated optimal trajectory π_P of a discrete sequence of configurations P. Each segment connecting subsequent configurations is optimal. Namely, its length equals the sub-Riemannian distance between two configurations.

associated optimal trajectory of P, denoted by $\pi_P : [0, S] \to \mathcal{X}$, is (i) dynamically feasible and (ii) sequentially connects the waypoints $\mathbf{q}_1, \ldots, \mathbf{q}_n$ such that each connection is locally optimal. Namely, it provides the shortest connection length $\ell(\pi_P(\mathbf{q}_i, \mathbf{q}_{i+1})) = d_{sr}(\mathbf{q}_i, \mathbf{q}_{i+1})$. See Fig. 4.

A key notion used in proving probabilistic exhaustivity is *tracing* (to be formally defined shortly). Roughly speaking, a given path x is said to be traced by a sequence of points P if they are close to x and the length of π_P, the path associated with P, is close to the length of the given path. This corresponds to ensuring a one-way Hausdorff distance between π_P and x ((iii) in Definition 5). In the original definition of (δ, r)-tracing, (see (Schmerling et al. 2015, Section IV)), requiring only one-way Hausdorff was sufficient as the proof was only concerned with path length (and not coverage). This is because if π_P "shortcuts" the original path, the path length is only reduced. However, in our setting, this may result in points covered by x not being covered by π_P. Thus, we add an additional requirement that all points along the path x are close to a point in P which, roughly speaking, corresponds to requiring the two-way Hausdorff distance ((iii) and (iv) in Definition 5).

Definition 5. (δ, r)-tracing. *Let $x : [0, T] \to \mathcal{X}$ be a dynamically feasible trajectory, $P = \{\mathbf{q}_i\}_{i=1}^n \subseteq \mathcal{X}$ be a discrete sequence of configurations and π_P be the associated optimal trajectory of P. P is said to $(\delta, r) - trace$ x if:*

(i) $d_{sr}\left(\mathbf{q}_i, \mathbf{q}_{i+1}\right)_{i=1,\ldots,n-1} \leq r$,

(ii) the arc length of π_P is bounded by $\ell(\pi_P) \leq (1 + \delta) \cdot \ell(x)$,

(iii) any point along the path π_P is at most r-distance away from some point along the path x. Namely, $\sup_{s\in[0,S]} \inf_{t\in[0,T]} d_{sr}\left(\pi_P(s), x(t)\right) \leq r$.

(iv) any point along the path x is at most r-distance away from some point in P. Namely, $\sup_{t\in[0,T]} \inf_{i\in\{1,\ldots n\}} d_{sr}\left(\mathbf{q}_i, x(t)\right) \leq r$.

See Fig. 5.

To use the extended (and slightly more restrictive) definition of tracing, we introduce the notion of an (r, k)-bounded trajectory. This notion formally defines to what extent a path can change (by limiting the curvature of a path) in a local neighborhood. This, in turn, will be used to inform us how many samples are required to trace a given path in order to prove probabilistic exhaustivity under our new definition of tracing.

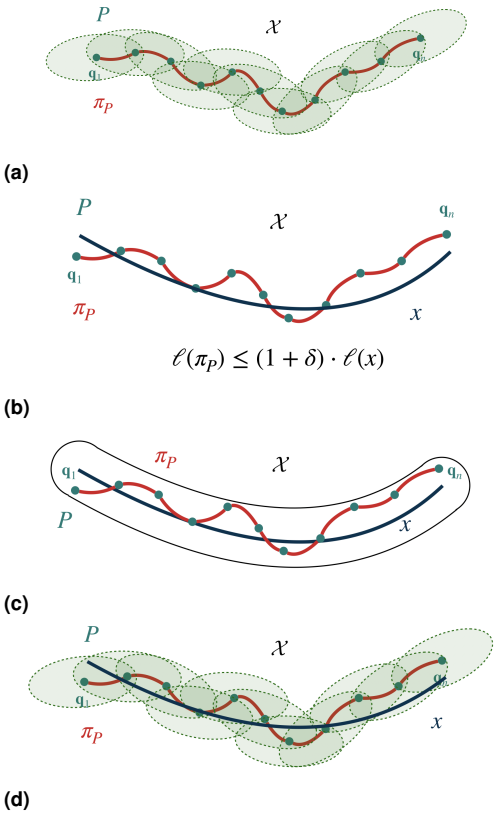


Figure 5. P (δ, r)-traces x. (a) The trajectory is dense enough, namely sequentially nearby configurations are within r sub-Riemaniian distance to another. (b) The arc length of the associated optimal trajectory is bounded. (c) π_P is close enough to x, in other words any point along π_P is at most r-distance away from some point along x. (d) x is close enough to P, in other words any point along x is at most r-distance away from some point in P.

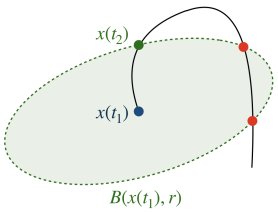


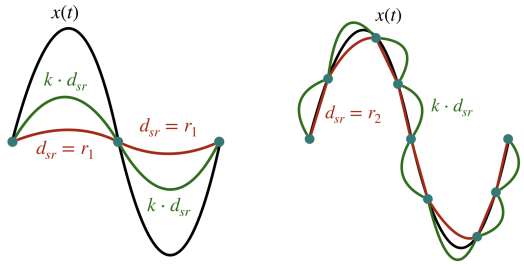
Figure 6. Depiction of t_2 used in Def. 6. There might be multiple points along x that satisfy $d_{sr}(x(t_1), x(t)) = r$ (the points on the boundary of the shown sub-Riemannian ball), we want t_2 to be the smallest.

Definition 6. (r_{curv}, k)-bounding. *Let $x : [0, T] \to \mathcal{X}$ be a dynamically feasible trajectory, for $k > 1$, x is said to be (r_{curv}, k)-bounded if $\forall r \in (0, r_{\text{curv}}]$, we have*

$$\ell\left(x(t_1, t_2)\right) \leq k \cdot d_{sr}(x(t_1), x(t_2)),$$
$$\forall\, 0 \leq t_1 < T,$$
$$t_2 = \min\left(\{t \in (t_1, T) : d_{sr}(x(t_1), x(t_2)) \geq r\} \cup \{T\}\right).$$

Here, $x(t_1, t_2)$ is the curve segment on x between $x(t_1)$ and $x(t_2)$. See Fig. 6.

We now prove that when the target trajectory is not pathological ($\|\dot{x}(t)\| < \infty$), then for any constant $k > 1$, we can always find a value r such that the trajectory is (r, k)-bounded. Intuitively, this will allow us to argue that



(a) $x(t)$ is not (r_1, k)-bounded. **(b)** $x(t)$ is (r_2, k)-bounded.

Figure 7. For a given $k > 1$, we can always find $r_{\text{curv}} > 0$ for which a trajectory x is (r_{curv}, k)-bounded. (a) An example of $x(t)$ not bounded when d_{sr} is too large. (b) An example of $x(t)$ bounded when d_{sr} small enough.

when we sample along a target trajectory at small-enough intervals (defined by r), the length of the curve connecting two nearby sampled points is not too long compared to the sub-Riemmanian distance between them.

Lemma 3. *Let $x : [0, T] \to \mathcal{X}$ be a dynamically feasible trajectory and $\dot{x}(t)$ is bounded, namely, $\forall t \in [0, T]$, $\|\dot{x}(t)\| \in [0, dx_{\max}]$. For every constant $k > 1$, there exists some radius $r = r_{\text{curv}}(k, x) > 0$ such that x is (r, k)-bounded. See Fig. 7.*

Proof. According to Definition 2, we have that

$$\ell(x(t_1, t_1 + \Delta t)) = \int_{t_1}^{t_1+\Delta t} \|\dot{x}(t)\| dt.$$

For certain t, if $\|\dot{x}(t)\| = 0$, it is straightforward that $d(\ell(x)) = 0$. When $0 < \|\dot{x}(t)\| \leq dx_{\max}$, take derivative for t on both sides

$$\frac{d(\ell(x))}{dt} = \|\dot{x}(t)\| \Rightarrow \frac{d(\ell(x))}{dt} = \frac{\|dx\|}{dt}$$
$$\Rightarrow d(\ell(x)) = \|dx\|$$

which can be written in another form as

$$\begin{aligned}
&\lim_{\Delta t \to 0} \left(\ell(x(0, t_1 + \Delta t)) - \ell(x(0, t_1))\right) \\
=&\lim_{\Delta t \to 0} \ell(x(t_1, t_1 + \Delta t)) \\
=&\lim_{\Delta t \to 0} \|x(t_1 + \Delta t) - x(t_1)\| \\
\leq&\lim_{\Delta t \to 0} d_{sr}(x(t_1), x(t_1 + \Delta t)) \\
<&\lim_{\Delta t \to 0} k \cdot d_{sr}(x(t_1), x(t_1 + \Delta t))
\end{aligned}$$

$$\lim_{\Delta t \to 0} \frac{\ell(x(t_1, t_1 + \Delta t))}{d_{sr}(x(t_1), x(t_1 + \Delta t))} = k_0(t_1) \leq 1 < k$$

According to the definition of limit, for some $\delta > 0$, there always exists $\xi > 0$ such that when $\Delta t - 0 < \xi$, $\frac{\ell(x(t,t+\Delta t))}{d_{sr}(x(t),x(t+\Delta t))} - k_0(t) < \delta$. Then we could take $\delta = k - 1$, then with corresponding ξ, take $r = \min\{d_{sr}(x(t), x(t + \xi)), t \in [0, T - \xi]\}$ would guarantee that x is (r, k)-bounded. ■

We now use Lemma 3 to prove that for a target trajectory x, given a large-enough number of samples, there exist a sequence of configurations that traces x, under the *new* definition of (δ, r)-tracing (Definition 5). To do so, we use several results that were used in the original proof of probabilistic exhaustivity.

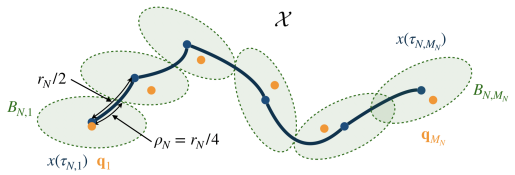


Figure 8. Sub-Riemannian balls (green) along x (dark blue) spaced at sub-Riemannian distances $r_N/2$. With high probability, each ball contains at least one point. We select one point from each of the balls to form a tracing sequence P_N (orange).

Lemma 4. Probabilistic exhaustivity. *Let $x : [0, T] \to \mathcal{X}_{\text{free}}$ be a dynamically feasible trajectory. Let $\mathcal{Q}_N$ be a set of N points sampled independently and identically from the uniform distribution on the collision-free space $\mathcal{X}_{\text{free}}$ and set $V_N = \{x(0)\} \cup \mathcal{Q}_N$. For a given N, set*

$$r_N = \kappa \cdot (\log(N)/N)^{1/D}.$$

*Here, D is a constant capturing the dimension of the system and κ is a commutable constant depending on the system dynamics, N, D, and some tuning parameter $\eta \geq 0$.** Let $\tilde{\mathcal{A}}_N$ be the event that there exists a discrete sequence of configurations $P = \{\mathbf{q}_i\}_{i=1}^n \subseteq V_N$ that (δ, r)-traces x for any $\delta \in (0, 1)$ and $r = r_N$. The probability that event $\tilde{\mathcal{A}}_N$ doesn't happen, denoted by $\mathbb{P}(\tilde{\mathcal{A}}_N^c)$ is asymptotically bounded by*

$$\mathbb{P}(\tilde{\mathcal{A}}_N^c) \leq O\left(N^{-\eta} \log^{-\frac{1}{D}} N\right).$$

As is defined above, $\eta \geq 0$ is some tuning parameter.

Proof. Define $\mathcal{T}_N = \{\tau_{N,1}, \tau_{N,M_N}\}$ as a sequence of points such that (i) $\tau_{N,1} = x(0)$, (ii) $x(\tau_{N,i})$ and $x(\tau_{N,i+1})$ are at sub-Riemannian distances $r_N/2$, where $\tau_{N,i+1} > \tau_{N,i}$ is the smallest timestamp that satisfies $d_{sr}(x(\tau_{N,i}), x(\tau_{N,i+1})) = r_N/2$ (see Fig. 6) and (iii) $\tau_{N,M_N} = x(T)$. Furthermore, define $B_N = \{B_{N,1}, B_{N,M_N}\}$ a sequence of sub-Riemannian balls such that $B_{N,i}$ is centered at $x(\tau_{N,i})$ and has radius $r_N/4$. Define $\mathcal{A}_N$ to be the event that each ball in B_N contains at least one point in V_N (see Fig. 8).

In (Schmerling et al. 2015, Thm. IV.5) it was shown that with probability at least

$$1 - O\left(N^{-\eta} \log^{-\frac{1}{D}} N\right)$$

it holds that there exists a discrete sequence of configurations $P = \{\mathbf{q}_i\}_{i=1}^n \subseteq V_N$ such that $\mathcal{A}_N$ exists and that requirements $(i) - (iii)$ in Definition 5 hold for δ and $r = r_N$. Thus, we only need to show that requirement (iv) in Definition 5 holds for δ and $r = r_N$. Namely, that

$$\sup_{t \in [0,T]} \inf_{i \in \{1,\ldots n\}} d_{sr}(\mathbf{q}_i, x(t)) \leq r.$$

Thus, we now show that $\forall t \in [0, T], \exists i$ s.t. $(\mathbf{q}_i, x(t)) \leq r$ which will conclude the proof. We consider two cases:

**For exact definitions of κ and D, see (Schmerling et al. 2015, Thm. IV.5).

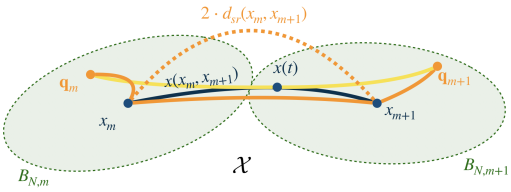


Figure 9. Shortest distances from arbitrary point along x to points in P_N. Sub-Riemannian distances are shown in orange and yellow. Trajectory segment $x(x_m, x_{m+1})$ is shown in dark blue.

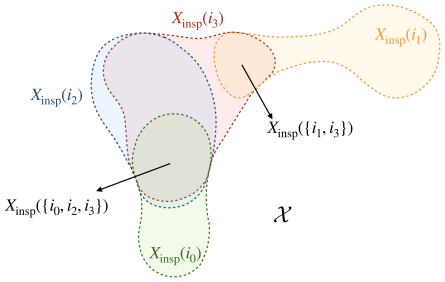


Figure 10. Inspecting configuration region (Definition 7).

C1 $x(t)$ lies at a center of some ball in B_N. Thus, there exists some i such that $x(t) = x(\tau_{N,i})$ and it is straightforward to see that $d_{\text{sr}}(\mathbf{q}_i, x(\tau_{N,i})) \leq r_N/4$.

C2 $x(t)$ does not lie at a center of some ball in B_N. Following Lemma 3, there exists some constant r_{curv} for which x is $(r_{\text{curv}}, 2)$-bounded. Fix N sufficiently large such that $r_N/2 \leq r_{\text{curv}}$. Such N exists as $\lim_{N\to\infty} r_N = 0$.

Assume that $x(t)$ lies on the curve connecting $x(\tau_{N,i})$ and $x(\tau_{N,i+1})$ for some i. We have that

$$\begin{aligned} & d_{sr}(x(\tau_{N,i}), x(t)) + d_{sr}(x(t), x(\tau_{N,i+1})) \\ & \leq \ell(x(\tau_{N,i}, t)) + \ell(x(t, \tau_{N,i+1})) \\ & = \ell(x(\tau_{N,i}, \tau_{N,i+1})) \\ & \leq 2 \cdot d_{sr}(x(\tau_{N,i}), x(\tau_{N,i+1})). \end{aligned} \tag{1}$$

The first inequality follows from Definition 3, the first equality follows from Definition 2, the second inequality follows from the fact that x is $(r_{\text{curv}}, 2)$-bounded.

Now,

$$\begin{aligned} & d_{sr}(\mathbf{q}_i, x(t)) + d_{sr}(x(t), \mathbf{q}_{i+1})) \\ & \leq d_{sr}(\mathbf{q}_i, x(\tau_{N,i})) + d_{sr}(x(\tau_{N,i}), x(t)) \\ & \quad + d_{sr}(x(t), x(\tau_{N,i+1})) + d_{sr}(x(\tau_{N,i+1}), \mathbf{q}_{i+1}) \\ & \leq d_{sr}(\mathbf{q}_i, x(\tau_{N,i})) + d_{sr}(\mathbf{q}_{i+1}, x(\tau_{N,i+1})) \\ & \quad + 2 \cdot d_{sr}(x(\tau_{N,i}), x(\tau_{N,i+1})) \\ & \leq r_N/4 + r_N/4 + 2 \cdot r_N/2 \\ & = (3/2) \cdot r_N \end{aligned}$$

the first inequality follows from Definition 3, the second inequality from Eq. (1), the third inequality follows from $d_{sr}(\mathbf{q}_i, x(\tau_{N,i})) \leq r_N/4$ and the fact that $x(\tau_{N,i})$ are spaced along x at sub-Riemannian distance $r_N/2$. See Fig. 9. Finally, we have

$$\begin{aligned} & \inf_{j\in\{1,\ldots n\}} d_{sr}(\mathbf{q}_j, x(t)) \\ & \leq \min\left(d_{\text{sr}}(\mathbf{q}_i, x(t)), d_{\text{sr}}(\mathbf{q}_{i+1}, x(t))\right) \\ & \leq (1/2) \cdot (3/2) \cdot r_N < r_N. \end{aligned}$$

■

B Well-behaved inspection trajectories

In this section, we introduce the notion of a *well-behaved trajectory*. Roughly speaking, it ensures that there are no singular points along a trajectory which a POI can only be inspected from. This, in turn, will allow us to ensure that trajectories that trace an optimal inspection path cover the same set of POI.

Recall that an inspection-planning problem $\mathcal{P} = (\mathcal{X}, \mathcal{I}, \mathcal{S}, \ell, q_s)$ is a tuple where $\mathcal{I}$ is the set of all POI, $\mathcal{X}$ is the configuration space of the robot and $\mathbf{q}_s$ is the start configuration. Furthermore, recall that $\mathcal{X}_{\text{free}} = \text{cl}(\mathcal{X}/\mathcal{X}_{\text{obs}})$ where $\text{cl}(\cdot)$ is the closure operator.

Definition 7. Inspecting configuration region. *Let $i \in \mathcal{I}$ be a point of interest (POI), the inspecting configuration region of i, denoted as $\mathcal{X}_{\text{insp}}(i)$, is defined to be the union of all configurations from which the POI can be inspected. Namely,*

$$\mathcal{X}_{\text{insp}}(i) = \{\mathbf{q} \in \mathcal{X}_{\text{free}} : i \in \mathcal{S}(\mathbf{q})\}.$$

Similarly, the inspecting configuration region of $\mathcal{I}' \subseteq \mathcal{I}$ is defined as

$$\mathcal{X}_{\text{insp}}(\mathcal{I}') = \{\mathbf{q} \in \mathcal{X}_{\text{free}} : \mathcal{I}' \subseteq \mathcal{S}(\mathbf{q})\}.$$

Definition 8. Well-behaving of an inspection trajectory. *Let $x : [0, T] \to \mathcal{X}_{\text{free}}$ be a feasible inspection trajectory. x is said to be strongly ξ-well behaved if $\forall i \in \mathcal{S}(x)$, there exists at least one point along x whose ξ-neighborhood is completely within the inspecting configuration region of i. Namely,*

$$\forall i \in \mathcal{S}(x), \exists t \in [0, T] \text{ s.t. } B^e(x(t), \xi) \subseteq \mathcal{X}_{\text{insp}}(i).$$

Similarly, x is said to be weakly ξ-well behaved if $\forall i \in \mathcal{S}(x) \setminus (\mathcal{S}(x(0)) \cup \mathcal{S}(x(T)))$, there exists at least one point along x whose ξ-neighborhood is completely within the inspecting configuration region of i. Namely,

$$\begin{aligned} & \forall i \in \mathcal{S}(x) \setminus (\mathcal{S}(x(0)) \cup \mathcal{S}(x(T))), \\ & \exists t \in [0, T] \text{ s.t. } B^e(x(t), \xi) \subseteq \mathcal{X}_{\text{insp}}(i). \end{aligned}$$

See also Fig. 11.

A strongly well-behaved trajectory extends into the inspecting configuration region of some POI while a weakly well-behaved trajectory can terminate at the boundary of some inspecting configuration region. It is not hard to see, that any strongly well-behaved trajectory can be shortened to a weakly well-behaved trajectory without losing coverage. Thus, we introduce the notion of *inspection completion time* which will simplify the proofs of the following lemmas and theorems.

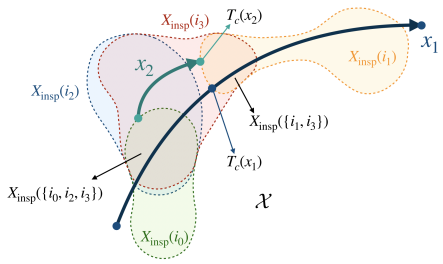


Figure 11. Strongly and weakly well-behaving trajectories (Definition 8). x_1 is strongly well-behaved, while x_2 is weakly well-behaved. For trajectory $x_1 : [0, T_1] \to \mathcal{X}$, inspection completion time $T_c(x_1) < T_1$. For trajectory $x_2 : [0, T_2] \to \mathcal{X}$, inspection completion time $T_c(x_2) = T_2$.

Definition 9. Inspection completion time. *Let $x : [0, T] \to \mathcal{X}_{\text{free}}$ be a feasible inspection trajectory, then the inspection completion time of x is defined as*

$$T_c(x) = \arg \min_{t \in [0,T]} \{\mathcal{S}(x(0,t)) = \mathcal{S}(x)\}.$$

Stated differently, the inspection completion time of x is the last point along x that adds a new POI covered by x. When it is clear from the context, we will use T_c instead of $T_c(x)$.

Using the above definitions, the following lemma gives the condition under which a discrete sequence of configurations covers all POI that are inspected along a continuous trajectory.

Lemma 5. *Let $x : [0, T] \to \mathcal{X}_{\text{free}}$ be a feasible weakly ξ-well behaved inspection trajectory and let $P = \{\mathbf{q}_m\}_{m=1}^n \subseteq \mathcal{X}_{\text{free}}$ be a discrete sequence of configurations. If*

(i) $\mathbf{q}_1 = x(0)$,

(ii) $\exists m \in [n]$ *such that* $\mathcal{S}(x(T_c)) \subseteq \mathcal{S}(\mathbf{q}_m)$, *and*

(iii) P (δ, r)*-traces* x *for* $r \leq \xi$,

then $\mathcal{S}(x) \subseteq \mathcal{S}(P)$.

Proof. The sequence P (δ, r)-traces x, thus following Definition 5, we have that

$$\sup_{t \in [0,T]} \inf_{j \in \{1,\dots n\}} \|\mathbf{q}_j - x(t)\| \leq \sup_{t \in [0,T]} \inf_{j \in \{1,\dots n\}} d_{sr}(\mathbf{q}_j, x(t)) \leq r.$$

Thus, $\forall t \in [0, T]$ we have that the closest point in P to $x(t)$, denoted by $\mathbf{q}_{\text{close}}(x, t)$ satisfies $\mathbf{q}_{\text{close}}(x, t) \in B^e(x(t), r)$. With $r \leq \xi$, we further have that $\mathbf{q}_{\text{close}}(x, t) \in B^e(x(t), r) \subseteq B^e(x(t), \xi)$.

Additionally, x is weakly ξ-well behaved. Thus following Definition 8, we have that

$$\forall i \in \mathcal{S}(x) \setminus (\mathcal{S}(x(0)) \cup \mathcal{S}(x(T))),$$
$$\exists t \in [0, T], j \in [n] \text{ s.t. } B^e(x(t), \xi) \subseteq \mathcal{X}_{\text{insp}}(i).$$

Namely, for every POI inspected along x (except for the endpoints of x), there exists at least one configuration in P from which the POI can also be inspected.

To show that $\mathcal{S}(x) \subseteq \mathcal{S}(P)$, we still need to show that $\mathcal{S}(x(0)) \subseteq \mathcal{S}(P)$ (which follows trivially from the fact that $\mathbf{q}_1 = x(0)$) and that $\mathcal{S}(x(T)) \subseteq \mathcal{S}(P)$. To show the latter, we consider two subcases:

C1 $T_c < T$. In this case $\mathcal{S}(x(T)) \subseteq \mathcal{S}(x)/\mathcal{S}(x(T))$, so $\mathcal{S}(x(T))$ is already covered by $\mathcal{S}(P)$,

C2 $T_c = T$. In this case $\mathcal{S}(x(T)) = \mathcal{S}(x(T_c)) \subseteq \mathcal{S}(\mathbf{q}_m)$ for some $m \in \{1, \dots, n\}$, so POI that is inspected at $x(T)$ is covered by such $\mathcal{S}(\mathbf{q}_m)$.

Thus, we have that $\mathcal{S}(x) \subseteq \mathcal{S}(P)$. ■

The following lemma gives the condition under which a continuous trajectory covers all POI inspected along another continuous trajectory.

Lemma 6. *Let $x : [0, T] \to \mathcal{X}_{\text{free}}$ be a feasible, weakly ξ-well behaved inspection trajectory with optimal coverage $\mathcal{S}(x) = \mathcal{I}^*$, and let $y : [0, T'] \to \mathcal{X}_{\text{free}}$ be a feasible inspection trajectory. If*

(i) $y(0) = x(0)$,

(ii) $y(T') = x(T)$, *and*

(iii) $\sup_{t \in [0,T]} \inf_{t' \in [0,T']} \|x(t) - y(t')\| \leq r$ *for* $0 < r < \xi$,

then $\mathcal{S}(y) = \mathcal{I}^*$ and y is weakly $(\xi - r)$-well behaved.

Proof. Following condition (iii) above, $\forall t \in [0, T]$ we have that the closest point in y to $x(t)$, denoted by $y_{\text{close}}(x, t)$ satisfies $y_{\text{close}}(x, t) \in B^e(x(t), r)$. As $r \leq \xi$, we further have that

$$y_{\text{close}}(x, t) \in B^e(x(t), \xi). \tag{2}$$

Additionally x is weakly ξ-well behaved. Thus, following Definition 8, we have that

$$\forall i \in \mathcal{S}(x), \exists t \in [0, T] \text{ s.t. } B^e(x(t), \xi) \subseteq \mathcal{X}_{\text{insp}}(i). \tag{3}$$

Combining Eq. 2 and 3, we have that

$$\forall i \in \mathcal{S}(x) \setminus (\mathcal{S}(x(0)) \cup \mathcal{S}(x(T))), \exists t, t' \in [0, T] \text{ s.t.,}$$
$$y(t') \in B^e(x(t), \xi) \text{ and } B^e(x(t), \xi) \subseteq \mathcal{X}_{\text{insp}}(i). \tag{4}$$

Eq. 4 implies that for every POI inspected along x (except for at the endpoints), there exists at least one configuration along y from which that POI can also be inspected. The POI covered at the endpoints, namely at $x(0)$ and $x(T)$, are covered by y following from conditions (i) and (ii) above. $\mathcal{I}^* = \mathcal{S}(x) \subseteq \mathcal{S}(y) \subseteq \mathcal{I}^*$, which, in turn, implies that $\mathcal{S}(y) = \mathcal{I}^*$.

We now show that y is also weakly $(\xi - r)$-well-behaved. Following Eq. (4) and the fact that $r < \xi$, for each $y(t') \in B^e(x(t), \xi)$, we further have that $B^e(y(t'), \xi - r) \subseteq B^e(x(t), \xi) \subseteq \mathcal{X}_{\text{insp}}$. This implies that $\forall i \in \mathcal{S}(x) \setminus (\mathcal{S}(x(0)) \cup \mathcal{S}(x(T)))$, there exists at least one point along y whose $(\xi - r)$-neighborhood is completely within the inspecting configuration region of i. Since

$$\mathcal{S}(x) \setminus (\mathcal{S}(x(0)) \cup \mathcal{S}(x(T))) = \mathcal{S}(y) \setminus (\mathcal{S}(y(0)) \cup \mathcal{S}(y(T'))),$$

and y satisfies the definition of being weakly $(\xi - r)$-well behaved. ■

C Asymptotic optimality of IRIS

In this section, we are finally ready to prove the asymptotic optimality of IRIS. To simplify the proof, we do so for a variant of IRIS where at each iteration we construct a PRM and not an RRG. While not identical, both roadmaps exhibit similar properties which are typically easier to show for PRMs (see, Karaman and Frazzoli (2011); Solovey et al. (2018)).

We start (Thm. 2) by showing that w.h.p. given a weakly ξ-well behaved trajectory with strong δ_{cl}-clearance and optimal inspection coverage, as the number of samples tends to infinity, IRIS returns an inspection trajectory whose length and coverage are bounded away from this trajectory. We then continue to introduce the notion of a *regular inspection problem* which, roughly speaking, states that the boundary of inspecting configuration regions has some structure. This will allow us (Thm. 3) to show that given a regular inspection planning problem, and an optimal inspection trajectory (that may be weakly ξ-well behaved, with weak δ_{cl}-clearance) then w.h.p. IRIS returns an inspection trajectory whose length and coverage are bounded away from this trajectory.

Theorem 2. IRIS comparison. *Let $\mathcal{P} = (\mathcal{X}, \mathcal{I}, \mathcal{S}, \ell, \mathbf{q}_s)$ be an inspection-planning problem and $\mathcal{X}_{\mathrm{free}}$ be the set of collision-free configurations. Assume that the robot system satisfies the assumptions mentioned in Schmerling et al. (2015) and let $x : [0, T] \to \mathcal{X}_{\mathrm{free}}$ be a feasible inspection trajectory such that*

(i) $x(0) = \mathbf{q}_s$,

(ii) x *has strong* δ_{cl}*-clearance for some* $\delta_{\mathrm{cl}} > 0$,

(iii) x *is weakly* ξ*-well behaved for some* $\xi > 0$,

(iv) x *has optimal coverage, namely* $\mathcal{S}(x) = \mathcal{I}^*$, *and*

(v) $\exists t \in [0, T]$ *such that* $B^e(x(t), \gamma) \subseteq \mathcal{X}_{\mathrm{insp}}(\mathcal{S}(x(T_c)))$ *for* $\gamma > 0$ *where* T_c *is the inspection completing point of* x.

Furthermore, let ℓ_N and $\mathcal{S}_N$ denote the arc length and inspection coverage, respectively, of the trajectory returned by IRIS at the N'th iteration using approximation parameters ε_N and p_N, and a lower-bound connecting radius r_N (as defined in Lemma 4). Then for any fixed $\delta \in (0, 1)$,

$$\mathbb{P}(\ell_N > (1+\varepsilon_N)(1+\delta)\ell(x)) = O\left(N^{-\eta}\log^{-\frac{1}{D}} N\right),$$

$$\mathbb{P}(|\mathcal{S}_N| < p_N \cdot |\mathcal{I}^*|) = O\left(N^{-\eta}\log^{-\frac{1}{D}} N\right).$$

Proof. Let $\mathcal{G}_N = (\mathcal{V}_N, \mathcal{E}_N)$ denote the roadmap constructed by IRIS at the N'th iteration and notice that $N = |\mathcal{V}|$. Assume N to be sufficiently large so that $r_N \leq \min\{\xi, \gamma, \delta_{\mathrm{cl}}/2\}$ and $\bigcup_{v \in \mathcal{V}_N} \mathcal{S}(v) = \mathcal{I}^*$.

Following Lemma 4, we have that with a probability of at least $1 - O\left(N^{-\eta}\log^{-\frac{1}{D}} N\right)$, there exists a discrete sequence of configurations $P = \{\mathbf{q}_m\}_{m=1}^n \subseteq \mathcal{V}$ that (δ, r_N)-traces x. We now show that in the event that such P exists, then IRIS will return a trajectory such that $\ell_N \leq (1+\varepsilon_N)(1+\delta)\ell(x)$ and $|\mathcal{S}_N| \geq p_N \cdot |\mathcal{I}^*|$.

Assume that such a sequence P exists. We can bound the minimal distance of any point in P to an obstacle in the configuration space as follows:

$$\begin{aligned}\forall m \inf_{a \in \mathcal{X}_{\mathrm{obs}}} \|\mathbf{q}_m - a\| &\geq \inf_{a \in \mathcal{X}_{\mathrm{obs}}} \|x - a\| - \|\mathbf{q}_m - x\| \\ &\geq \delta_{\mathrm{cl}} - r_N \\ &\geq r_N.\end{aligned}$$

Thus, $B^e(\mathbf{q}_m, r_N) \subseteq \mathcal{X}_{\mathrm{free}}$. As $P = \{\mathbf{q}_m\}_{m=1}^n$ (δ, r_N)-traces x, for any point $\mathbf{q}$ along the locally optimal edge between $\mathbf{q}_m$ and $\mathbf{q}_{m+1}$, we have

$$\|\mathbf{q}_m - \mathbf{q}\| \leq d_{sr}(\mathbf{q}_m, \mathbf{q}) \leq d_{sr}(\mathbf{q}_m, \mathbf{q}_{m+1}) \leq r_N.$$

Thus $\mathbf{q} \in B^e(\mathbf{q}_m, r_N) \subseteq \mathcal{X}_{\mathrm{free}}$. Namely, all the vertices of P as well as locally optimal paths connecting subsequent vertices in P are collision-free. This, together with the connection radius used ensures that all edges connecting vertices along P are collision-free and are added to $\mathcal{E}_N$.

By construction, we have that $\mathbf{q}_1 = \mathbf{q}_s = x(0)$. In addition, following condition (v) and Definition 5 we have that $\exists m \in [n]$ such that $\mathcal{S}(\mathbf{q}_m) = \mathcal{S}(x(T_c))$. Together with the fact that x is weakly ξ-well behaved, P (δ, r_N)-traces x, and $r_N \leq \xi$ then by Lemma 5 it holds that $\mathcal{S}(x) \subseteq \mathcal{S}(P)$. As x has optimal coverage (condition (iv)), we have that $\mathcal{S}(P) = \mathcal{I}^*$.

Let ℓ_N^* and $\mathcal{S}_N^*$ denote the arc length and inspection coverage of optimal inspection trajectory on roadmap $\mathcal{G}_N$, respectively. We have that

$$\begin{aligned}\ell_N \leq (1+\varepsilon_N)\ell_N^* &\leq (1+\varepsilon_N)\ell(\pi_P) \\ &\leq (1+\varepsilon_N)(1+\delta)\ell(x).\end{aligned}$$

Here, the first inequality follows from Thm. 1, the second inequality follows from the fact that ℓ_N^* is the infimum taken over the arc length of all trajectories that has optimal coverage on the graph, and the last inequality follows from Definition 5.

Similarly, we have that

$$|\mathcal{S}_N| \geq p_N \cdot |\mathcal{S}_N^*| = p_N \cdot |\mathcal{I}^*|.$$

Here, the first inequality follows from Thm. 1 and the first equality follows from the fact that $\mathcal{S}_N^* = \mathcal{I}^*$. ■

Before stating our final result, we introduce the notion of a regular boundary and regular inspection planning problem. This is required because an optimal inspection trajectory will never be strongly well-behaved but only weakly well-behaved and special care needs to be taken in order for IRIS to cover the POI covered by $x^*(T)$. The notion of regularity will ensure that there always exists a region near $x^*(T)$ that IRIS can sample inside. See also Fig. 12.

Definition 10. Regular boundary. *A set $\mathcal{X}' \subseteq \mathcal{X}_{\mathrm{free}}$ is said to have a regular boundary if there exists $\gamma > 0$ such that $\forall \mathbf{q} \in \partial\mathcal{X}$, there exists $\mathbf{q}' \in \mathcal{X}$ with $B^e(\mathbf{q}', \gamma) \subseteq \mathcal{X}'$ and $\mathbf{q} \in \partial B^e(\mathbf{q}', \gamma)$.*

Definition 11. Regularity of an inspection-planning problem. *Let $\mathcal{P} = (\mathcal{X}, \mathcal{I}, \mathcal{S}, \ell, q_s)$ be an inspection-planning problem and $\mathcal{X}_{\mathrm{free}}$ be the set of collision-free configurations. $\mathcal{P}$ is said to be regular if $\forall \mathbf{q} \in \mathcal{X}_{\mathrm{free}}$, $\mathcal{X}_{\mathrm{insp}}(\mathcal{S}(\mathbf{q}))$ has a regular boundary.*

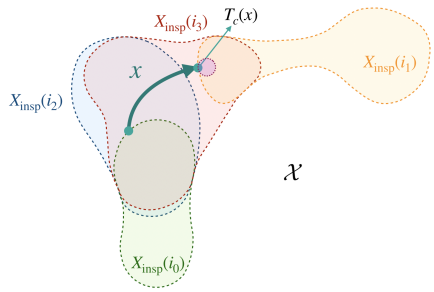


Figure 12. If the inspection planning problem is regular, namely all (non-empty) inspecting configuration regions have regular boundaries, then there always exists a small region near $x(T)$ that is within the same inspecting configuration region (denoted by purple circle in the figure).

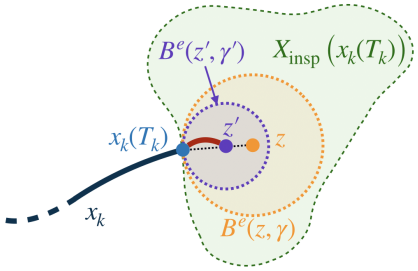


Figure 13. We construct an extended trajectory x'_k by extending x_k into $\mathcal{X}_{\text{insp}}(x_k(T_k))$. The extended part is shown in red, whose length is required to be shorter than $(\Delta/4)\ell(x^*)$.

Definition 12. Strong/weak δ_{cl}-clearance. *Let $x:[0,T]\to\mathcal{X}_{\text{free}}$ be a feasible trajectory. x has strong δ_{cl}-clearance if $\forall t\in[0,T]$, $x(t)$ is in δ_{cl}-interior of $\mathcal{X}_{\text{free}}$ (namely, $x(t)$ is at least δ_{cl} away from any point in $\mathcal{X}_{\text{obs}}$ using the Euclidean distance). Furthermore, x has weak δ_{cl}-clearance if there exists a sequence of homotopic paths $\{x_k\}_{k\in\mathbb{N}}$ that satisfies:*

(i) $\lim_{k\to\infty} x_k = x$.

(ii) x_0 *has strong* δ_{cl}*-clearance.*

(iii) $\forall k\in[0,\infty)$, x_k *is dynamically feasible and has strong* δ_k*-clearance for some* $\delta_k>0$*, and* $\lim_{k\to\infty}\delta_k = 0$.

(iv) $\lim_{k\to\infty}\ell(x_k)=\ell(x)$.

We are now ready to state our final result.

Theorem 3. IRIS asymptotic optimality. *Let $\mathcal{P}=(\mathcal{X},\mathcal{I},\mathcal{S},\ell,\mathbf{q}_s)$ be a regular inspection-planning problem and $\mathcal{X}_{\text{free}}$ be the collision-free space. Assume that the robot system satisfies the assumptions mentioned in Schmerling et al. (2015) and let $x^*:[0,T]\to\mathcal{X}_{\text{free}}$ be an optimal feasible inspection trajectory such that*

(i) $x^*(0)=\mathbf{q}_s$,

(ii) x^* *has weak* δ_{cl}*-clearance for some* $\delta_{\text{cl}}>0$,

(iii) x^* *is weakly* ξ*-well behaved for some* $\xi>0$.

Furthermore, let ℓ_N and $\mathcal{S}_N$ denote the arc length, and the inspection coverage, respectively of the trajectory returned by IRIS at the N'th iteration using approximation parameters ε_N and p_N, and a lower-bound connecting radius r_N (as defined in Lemma 4). If $\lim_{N\to\infty}\varepsilon_N=0$ and $\lim_{N\to\infty}p_N=1$, we have that

$$\lim_{N\to\infty}\mathbb{P}(\ell_N>(1+\Delta)\ell(x^*))=0$$

for any $\Delta>0$ and that

$$\lim_{N\to\infty}\mathbb{P}(|\mathcal{S}_N|<|\mathcal{I}^*|)=0.$$

Proof. Assume w.l.o.g. that $\ell(x^*)>0$. Following (Karaman and Frazzoli 2011, Lemma 50) and the fact that x^* has weak δ_{cl}-clearance, there exists a sequence $\{x_k\}_{k\in\mathbb{N}}$ of paths such that $\lim_{k\to\infty}x_k=x^*$ and x_k has strong δ_k-clearance where $\{\delta_k\}_{k\in\mathbb{N}}$ is a sequence of real numbers such that $\lim_{k\to\infty}\delta_k=0$.

Choose $k_0\in\mathbb{N}$ such that $\forall k\geq k_0$ we have that

$$\ell(x_k)\leq(1+\Delta/4)\cdot\ell(x^*),$$

and that

$$\sup_{t^*\in[0,T^*]}\inf_{t\in[0,T]}\|x_k(t)-x^*(t^*)\|\leq\xi-\xi_k,$$

with $\xi_k\in(0,\xi)$. Furthermore, we assume that k_0 is large enough such that $\forall k\geq k_0$, the inspection completing point of x_k equals its completion time (namely, $T_c(x_k)=T_k$ where T_k is the completion time of x_k).

Notice that as x^* and x_k are homotopic, we have that $x_k(0)=x^*(0)$ and $x_k(T_k)=x^*(T)$. Thus, using Lemma 6 we have that $\mathcal{S}(x_k)=\mathcal{I}^*$ and that x_k is weakly ξ_k-well behaved.

To use Thm. 2, we show how to construct a new trajectory x'_k that extends x_k into the interior of $\mathcal{X}_{\text{insp}}(\mathcal{S}(x_k(T_c)))$. This is similar to the construction used in (Schmerling et al. 2015, Thm. VI.2). As $x_k(T_k)\in\partial\mathcal{X}_{\text{insp}}(x_k(T_k))$ and since the inspection planning problem $\mathcal{P}$ is regular, then there exists some configuration z such that $z\in\mathcal{X}_{\text{insp}}(x_k(T_k))$, $B^e(z,\gamma)\subseteq\mathcal{X}_{\text{insp}}(x_k(T_k))$ and $x^*(T)\in\partial B^e(z,\gamma)$ for some $\gamma>0$ (see also Fig. 13).

Set z' to be the configuration on the straight-line segment connecting $x_k(T_k)$ and z such that $d_{sr}(x_k(T_k),z')\leq(\Delta/4)\ell(x^*)$. Finally, set x'_k to be the extension of x_k constructed by concatenating x_k with the shortest sub-Riemannian path between $x_k(T_k)$ and z'. We now bound the arc length of x'_k by

$$\begin{aligned}\ell(x'_k)&\leq\ell(x_k)+d_{sr}(x_k(T_k),z')\\&\leq\ell(x_k)+(\Delta/4)\ell(x^*)\\&\leq(1+\Delta/4)\cdot\ell(x^*)+(\Delta/4)\cdot\ell(x^*)\\&=(1+\Delta/2)\cdot\ell(x^*).\end{aligned}\tag{5}$$

Clearly, for $\gamma'=\|z'-x_k(T)\|$ it holds that $B^e(z',\gamma')\subseteq B^e(z,\gamma)\subseteq\mathcal{X}_{\text{insp}}(x_k(T_k))$. In addition, for any point $p\in x'_k$ along the path between $x_k(T_k)$ and z' we have that

$$\begin{aligned}\inf_{a\in\mathcal{X}_{\text{obs}}}\|p-a\|&\geq\inf_{a\in\mathcal{X}_{\text{obs}}}\|x_k(T_k)-a\|-\|p-x_k(T_k)\|\\&\geq\delta_k-\delta_k/2\\&=\delta_k/2.\end{aligned}$$

Thus, x'_k has strong $\delta_k/2$ clearance.

Notice that x'_k satisfies all requirements of Thm. 2. Using a value of $\delta := \Delta/4$, we have that

$$\begin{aligned}
&O\left(N^{-\eta}\log^{-\frac{1}{D}}N\right)\\
=&\mathbb{P}(\ell_N > (1+\varepsilon_N)(1+\Delta/4)\cdot\ell(x'_k))\\
\geq&\mathbb{P}(\ell_N > (1+\varepsilon_N)(1+\Delta/4)(1+\Delta/2)\cdot\ell(x^*))\\
\geq&\mathbb{P}(\ell_N > (1+\varepsilon_N)(1+\Delta)\cdot\ell(x^*)).
\end{aligned}$$

Here, the first equality follows from Thm. 2, the first inequality follows from equation (5), and the last inequality follows from the fact that $\Delta \in (0,1)$. Similarly, Thm. 2 implies that

$$\mathbb{P}(|\mathcal{S}_N| < p_N \cdot |\mathcal{I}^*|) = O\left(N^{-\eta}\log^{-\frac{1}{D}}N\right).$$

Using the fact that $\lim_{N\to\infty}\varepsilon_N = 0$ and that $\lim_{N\to\infty} p_N = 1$, we have that

$$\begin{aligned}
0 &= \lim_{N\to\infty} O\left(N^{-\eta}\log^{-\frac{1}{D}}N\right)\\
&= \lim_{N\to\infty} \mathbb{P}(\ell_N > (1+\varepsilon_N)(1+\delta)\ell(x^*))\\
&= \lim_{N\to\infty} \mathbb{P}(\ell_N > (1+0)(1+\delta)\ell(x^*))\\
&= \lim_{N\to\infty} \mathbb{P}(\ell_N > (1+\delta)\ell(x^*)).
\end{aligned}$$

Similarly,

$$\begin{aligned}
0 &= \lim_{N\to\infty} O\left(N^{-\eta}\log^{-\frac{1}{D}}N\right)\\
&= \lim_{N\to\infty} \mathbb{P}(|\mathcal{S}_N| < p_N \cdot |\mathcal{I}^*|)\\
&= \lim_{N\to\infty} \mathbb{P}(|\mathcal{S}_N| < 1 \cdot |\mathcal{I}^*|)\\
&= \lim_{N\to\infty} \mathbb{P}(|\mathcal{S}_N| < |\mathcal{I}^*|)
\end{aligned}$$

■