2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

# GuILD: Guided Incremental Local Densification for Accelerated Sampling-based Motion Planning

Rosario Scalise*[1], Aditya Mandalika*[2], Brian Hou*[1], Sanjiban Choudhury[3] and Siddhartha S. Srinivasa[1]

***Abstract*— Sampling-based motion planners rely on incremental densification to discover progressively shorter paths. After computing feasible path $\xi$ between start $x_s$ and goal $x_t$, the *Informed Set* (IS) prunes the configuration space $\mathcal{X}$ by conservatively eliminating points that cannot yield shorter paths. Densification via sampling from this Informed Set retains asymptotic optimality of sampling from the entire configuration space. For path length $c(\xi)$ and Euclidean heuristic $h$, $IS = \{x | x \in \mathcal{X}, h(x_s, x) + h(x, x_t) \leq c(\xi)\}$.**

**Relying on the heuristic can render the IS especially conservative in high dimensions or complex environments. Furthermore, the IS only shrinks when shorter paths are discovered. Thus, the computational effort from each iteration of densification and planning is wasted if it fails to yield a shorter path, despite improving the cost-to-come for vertices in the search tree. Our key insight is that even in such a failure, shorter paths to *vertices in the search tree* (rather than just the goal) can immediately improve the planner's sampling strategy. Guided Incremental Local Densification (`GuILD`) leverages this information to sample from *Local Subsets* of the IS. We show that `GuILD` significantly outperforms uniform sampling of the Informed Set in simulated $\mathbb{R}^2$, $SE(2)$ environments and manipulation tasks in $\mathbb{R}^7$.**

## I. Introduction

Sampling-based algorithms have shown tremendous success in solving complex high-dimensional robot motion planning problems. To achieve asymptotic-optimality, these algorithms incrementally sample the robot's configuration space to continually improve the shortest path in an anytime manner [1]–[6]. A discrete sampling-based approximation to the continuous configuration space often yields an initial feasible path quickly. However, converging to the optimal path is typically slow: many configuration samples are needed to improve the discrete approximation (i.e., cover the space more densely) and discover a shorter path. Replanning with new samples can be accomplished efficiently [7], so the computational bottleneck is sampling in regions that result in a better path.

Our focus is on accelerating convergence to the optimal path, assuming that an initial feasible path has already been discovered. The current best path $\xi$ between the start $x_\mathrm{s}$ and target $x_\mathrm{t}$ defines an *Informed Set* $\mathcal{X}_\mathrm{IS} \subseteq \mathcal{X}$ that contains only states that can yield shorter paths:

$$\mathcal{X}_\mathrm{IS} = \{x \mid x \in \mathcal{X}, h(x_\mathrm{s}, x) + h(x, x_\mathrm{t}) \leq c(\xi)\}, \quad (1)$$

*Authors contributed equally.

[1] Paul G. Allen School of Computer Science and Engineering, University of Washington. [2] Aurora Innovation Inc. [3] Cornell University

This work was (partially) funded by the National Science Foundation IIS (#2007011), National Science Foundation DMS (#1839371), the Office of Naval Research, DEVCOM Army Research Laboratory, Amazon, and Honda Research Institute USA.

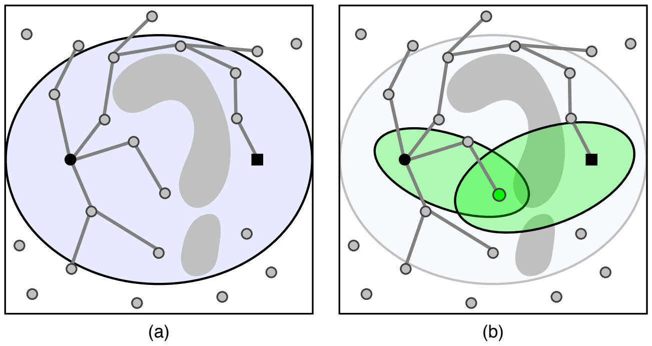


Fig. 1: Comparison of (a) Informed Set and (b) *Local Subsets* induced by `GuILD`. To further focus sampling within the Informed Set, `GuILD` chooses a beacon (green) and decomposes the original problem into two smaller problems using information from the search tree.

where $c(\xi)$ is the cost of the current path and $h$ is an admissible heuristic [8]. For problems minimizing path length in $\mathbb{R}^n$, the Informed Set with the Euclidean distance heuristic is an $n$-dimensional prolate hyperspheroid $\mathcal{E}_\mathrm{IS}$, parameterized by foci $x_\mathrm{s}$ and $x_\mathrm{t}$ and transverse axis diameter $c(\xi)$ (i.e., a generalized ellipse as seen in Fig. 1a). Sampling from $\mathcal{E}_\mathrm{IS}$ preserves the asymptotic optimality guarantee of sampling from the entire configuration space $\mathcal{X}$.

In practice, however, the Informed Set often provides limited sample efficiency improvement. Since the measure ($n$-volume) of the Informed Set $\lambda(\mathcal{E}_\mathrm{IS})$ is a function of the current solution cost, a high cost solution may yield a large Informed Set with comparable (or greater) measure to the full state space $\lambda(\mathcal{X})$. Paradoxically, to reap the greatest benefits of sampling from the Informed Set, the planner must already have a path of sufficiently small cost. This is a particular challenge for planning problems in high dimensions or cluttered environments where feasible paths likely have high cost. Furthermore, the computational effort from each iteration of densification and planning is wasted if it fails to yield a shorter path. Until a shorter path is discovered, $\mathcal{E}_\mathrm{IS}$ (and thus the sampling distribution) remains unchanged.

Our key insight is that even in such a failure, we can open the black box of the planning algorithm to immediately improve the planner's sampling strategy. With **Gu**ided **I**ncremental **L**ocal **D**ensification (`GuILD`), shorter paths to *any vertex in the search tree* can guide further sampling. `GuILD` introduces the idea of a beacon, a vertex in the search tree that decomposes the original sampling/planning problem into two smaller subproblems (Fig. 1b). Much like the Informed Set between the start and target, the beacon induces

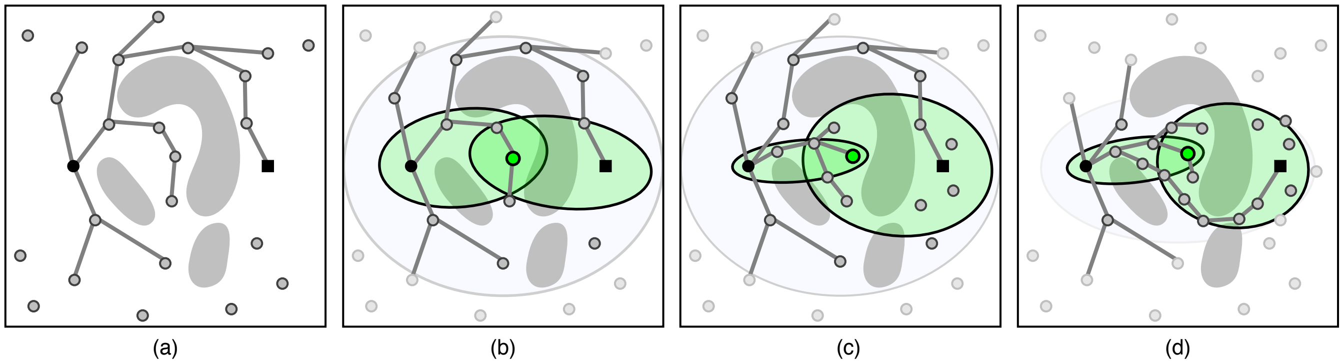


Fig. 2: **(a)** Initial solution. **(b)** `GuILD` selects a beacon and induces Local Subsets (green), which do not cover the narrow passage. **(c)** The planner does not find a shorter path to the goal, so the Informed Set is unchanged. However, `GuILD` leverages improved cost-to-come in the search tree to update Local Subsets. The start-beacon set shrinks to further focus sampling, and the remaining slack between beacon's and goal's cost-to-comes *expands* the beacon-target set (described in detail in section IV). Local Subsets now cover the narrow passage, focusing sampling within the Informed Set to quickly converge to **(d)** the optimal solution.

*Local Subsets* with (i) start and beacon as foci and (ii) beacon and target as foci. `GuILD` leverages improvements to the search tree to adapt the Local Subsets and converge to the optimal path with fewer samples (Fig. 2).

Generating samples from the continuous configuration space that will accelerate convergence to the optimal path is a challenging problem. Most approaches extract geometric information or low-dimensional structure from the robot workspace to bias the sampling distribution [9]–[12]. However, these methods can be computationally expensive and sensitive to covariate shift. `GuILD` simplifies this into a discrete problem of identifying beacons that yield focused samples. Local Subsets can be sampled efficiently, and the distribution is biased online by leveraging cost information already captured in the search tree.

`GuILD` only uses vertices in the search tree to construct Local Subsets. While any configuration in the Informed Set can be used to define subproblems, the lower bounds from the admissible heuristic are insufficient for constraining the two transverse axis diameters. In contrast, the current cost-to-come for vertices in the search tree provides an additional upper bound that uniquely defines the Local Subsets.

We make the following contributions:

- We introduce `GuILD`, an incremental densification framework that effectively leverages search tree information to focus sampling.
- We compare theoretical properties of the Local Subsets that `GuILD` samples to the Informed Set.
- We propose several BEACONSELECTOR strategies for `GuILD`, including an adversarial bandit algorithm.
- We show experimentally that regardless of the BEACONSELECTOR, `GuILD` outperforms the state-of-the-art Informed Set densification baseline across a range of planning domains. In particular, `GuILD` yields modest improvements in simpler planning domains and excels in domains with difficult-to-sample homotopy classes.

## II. RELATED WORKS

Sampling-based algorithms construct roadmaps [13] or trees [14]–[16] by sampling configurations and connecting nearest neighbors to compute a collision-free shortest path. Sampling-based algorithms for *optimal* motion planning incrementally densify the configuration space to determine shorter paths in an anytime manner [1], [3]–[6], [17] guaranteeing asymptotic optimality. However, their convergence is typically slow, especially in higher dimensions and complex environment with narrow passages; a uniform sampler requires $O(\delta^{-d})$ configuration samples to discover the optimal path with $\delta$-clearance in a $d$-dimensional space, which can be computationally prohibitive [16].

This motivated literature that focused on improving the sampling strategy. A popular approach is to extract information from the workspace to bias sampling. One class of algorithms samples *between* regions of collision to identify narrow passages [9], [18]–[23], or samples close to obstacles to compute high quality paths that follow contours of the obstacles [24]–[26]. This reduces the number of samples required to compute a solution, at the expense of computationally-intensive geometric tests to accept samples, resulting in large planning times. Another class of algorithms [27], [28] instead bias sampling by initially solving a coarse approximation of the workspace, and the solution informs subsequent sampling. These approaches, however, can be used in parallel with `GuILD` to sample non-uniformly within the Local Subsets.

Recently, to alleviate the issues with approaches relying on explicit geometric tests, there has been effort in determining low-dimensional structure in the workspace to bias sampling. Generative modeling tools have been applied to learn sampling distributions, conditioned upon the current planning problem and the obstacle distribution [10]–[12], [29]–[31]. These approaches do not explicitly reason about the Informed Set to provide optimality guarantees, and fail to robustly sample bottleneck regions in complex environments. As a result, the benefits are limited to environments that are similar to those reflected in the training data. In contrast, GuILD makes no assumption about the environment structure. However, these learning-based approaches share with GuILD the underlying idea of identifying critical samples and can be leveraged to inform beacon selection.

A different class of algorithms adapt sampling online as planning progresses. GuILD best fits in this class of algorithms that bias sampling with the state of the planning algorithm. Toggle-PRM [32] constructs one roadmap in the free space and another in the obstacle space to infer narrow passages and increase sampling density in such regions, but does not adapt in a cost-sensitive manner. Some approaches trade-off between exploration of the configuration space and exploitation of the underlying cost space [33]. DRRT [34] guides sampling by following a gradient of the underlying cost function. However, these algorithms require additional information from the cost function; GuILD assumes the cost function is a black box to be evaluated. Guided-EST [35] and Relevant Regions [36]–[38] are *tree*-based algorithms that bias sampling within the Informed Set by considering cost and the current local sampling density. In this work, GuILD focuses on sampling increasingly dense *roadmaps* via incremental densification, and thus algorithms such as BIT* [4] are most appropriate for comparison.

## III. Sampling-Based Optimal Motion Planning via Incremental Densification

In this section, we formally introduce the problem of sampling-based optimal motion planning (optimal SBMP). Let $\mathcal{X} \subseteq \mathbb{R}^n$ be the statespace of the planning problem, where $\mathcal{X}_{\text{obs}} \subset \mathcal{X}$ is the subspace occupied by obstacles and the free space is $\mathcal{X}_{\text{free}} = \mathcal{X} \backslash \mathcal{X}_{\text{obs}}$. Given source and target states $x_{\text{s}}, x_{\text{t}} \in \mathcal{X}_{\text{free}}$, a path $\xi : [0,1] \rightarrow \mathcal{X}$ is represented as a sequence of states such that $\xi(0) = x_{\text{s}}$ and $\xi(1) = x_{\text{t}}$. Let $\Xi(x_{\text{s}}, x_{\text{t}})$ be the set of all such paths. Given a cost function $c : \xi \rightarrow \mathbb{R}_{\geq 0}$, an optimal collision-free path is defined as:

$$\xi^* = \underset{\xi \in \Xi(x_{\text{s}}, x_{\text{t}})}{\arg\min} \; c(\xi) \quad s.t. \quad \forall t \in [0,1], \; \xi(t) \in \mathcal{X}_{\text{free}}. \quad (2)$$

Optimal SBMP algorithms progressively improve a roadmap approximation of the state space to plan asymptotically-optimal paths (Alg. 1). In each iteration, states are sampled from $\mathcal{X}_{\text{free}}$ to grow the vertices of an edge-implicit[1] graph $\mathcal{G}$ (Line 4). Then, a shortest path algorithm computes the *resolution-optimal* path on $\mathcal{G}$, internally using an admissible heuristic $h$ to focus the search (Line 5). If densification produces a lower-cost path, the best solution cost is updated and that path is emitted as $\xi_i$ (Lines 6-8). As the discrete graph $\mathcal{G}$ more closely approximates the continuous state space, the sequence of resolution-optimal paths $\{\xi_1, \xi_2, \cdots\}$ approaches the optimal path $\xi^*$ in cost.

### *Informed Incremental Densification*

Given candidate state $x \in \mathcal{X}_{\text{free}}$ and admissible heuristic $h$, the cost of all paths between $x_{\text{s}}$ and $x_{\text{t}}$ that pass through $x$ is lower bounded by $h(x_{\text{s}}, x) + h(x, x_{\text{t}})$. Defined by Eq. 1, the Informed Set (IS) excludes states for which this lower bound exceeds the best solution cost $c(\xi)$. Unlike sampling the entire state space $\mathcal{X}_{\text{free}}$, sampling from the IS automatically excludes states that cannot be part of a lower-cost path [8].

[1]The radius of connectivity of the implicit graph is chosen to ensure asymptotic optimality [1].

**Algorithm 1:** Informed Optimal SBMP

```
  Input: start $v_{\text{s}}$, goal $v_{\text{t}}$
  Output: $\{\xi_1, \; \xi_2, \cdots\}$ s.t. $c(\xi_{i+1}) < c(\xi_i)$
1 Initialize best solution cost: $c(\xi) \leftarrow \infty$
2 Initialize edge-implicit graph: $\mathcal{G} \leftarrow \{v_{\text{s}}, v_{\text{t}}\}$
3 repeat
4     Densify: $\mathcal{G} \xleftarrow{+}$ Sample($v_{\text{s}}$, $v_{\text{t}}$, $c(\xi)$)      $(\star)$
5     Compute shortest path: $\hat{\xi} \leftarrow$ Search($v_{\text{s}}$, $v_{\text{t}}$, $\mathcal{G}$)
6     if $c(\hat{\xi}) = g(v_{\text{t}}) < c(\xi)$ then
7         Update best solution cost: $c(\xi) \leftarrow g(v_{\text{t}})$
8         Emit current solution $\hat{\xi}$
9 until forever;
```

Euclidean distance, an admissible heuristic for path length, admits a concise geometric description of the IS: an $n$-dimensional prolate hyperspheroid $\mathcal{E}_{\text{IS}} = \mathcal{E}(x_{\text{s}}, x_{\text{t}}, c(\xi))$ with foci $x_{\text{s}}, x_{\text{t}}$ and transverse axis diameter $c(\xi)$.

$$\mathcal{E}_{\text{IS}} = \{x \mid x \in \mathcal{X}, \|x_{\text{s}} - x\|_2 + \|x - x_{\text{t}}\|_2 \leq c(\xi)\} \quad (3)$$

Prolate hyperspheroids can be sampled analytically, rather than via rejection sampling [8].

Sampling from the IS is a sufficient condition to converge to $\xi^*$. The improved efficiency can be characterized by comparing the measure ($n$-volume) $\lambda(\mathcal{E}_{\text{IS}})$ to the measure of the full state space $\lambda(\mathcal{X})$. When $\lambda(\mathcal{E}_{\text{IS}}) < \lambda(\mathcal{X})$, the IS can yield significant improvements on sample efficiency. However, when the initial solution has high path length (e.g., due to a cluttered environment), $\lambda(\mathcal{E}_{\text{IS}})$ may instead be closer to—or even larger than—$\lambda(\mathcal{X})$. This issue is exacerbated because the IS only shrinks when shorter paths are found. As a result, each iteration of densification and search that fails to find a shorter path does not affect the state sampling distribution. In the next section, we introduce a new strategy that leverages this previously-wasted computational effort.

## IV. Guided Incremental Local Densification

We present GuILD, an incremental densification framework that leverages partial search information to focus sampling within the IS. If an iteration of densification and planning has failed to improve the solution cost, what information is there for GuILD to take advantage of?

Our key insight is to open the black box of the underlying search algorithm. Although the iteration may not have found a shorter path to the goal, new shorter paths to *other vertices in the search tree* can immediately improve the sampling strategy (Alg. 1, Line 4).

### *Guiding Densification with Search Tree Information*

During search (Alg. 1, Line 5), the algorithm internally expands vertices in $\mathcal{G}$ to construct a search tree $\mathcal{T}$. Each vertex $v \in \mathcal{T}$ is associated with a parent vertex, a cost-to-come $g(v)$ via that parent, and a consistent cost-to-go

**Algorithm 2:** `GuILD`

```
    Input: beacon set $\mathcal{B}$, search tree $\mathcal{T}$, graph $\mathcal{G}$
    Output: densified graph $\mathcal{G}$
1   if a solution does not yet exist then
2       $\mathcal{G} \xleftarrow{+}$ UniformSample($\mathcal{X}_{\text{free}}$)
3   else
4       $b \leftarrow$ BeaconSelector($\mathcal{B}$)
5       $\mathcal{E}_{\text{LS}} \leftarrow \mathcal{E}(v_s, b, g(b)) \cup \mathcal{E}(b, v_t, c(\xi) - g(b))$
6       $\mathcal{G} \xleftarrow{+}$ UniformSample($\mathcal{E}_{\text{LS}}$)
7   return $\mathcal{G}$
```

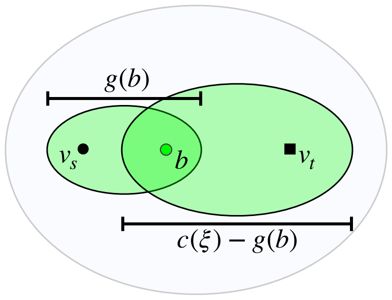


Fig. 3: Local Subsets $\mathcal{E}_{\text{LS}}$ (green) are defined by a beacon $b$, as well as the cost-to-come on the search tree $g(b)$ and the current best solution cost $c(\xi)$.

heuristic $h(v, v_t)$. When new states are sampled and added to $\mathcal{G}$, they may also be added to $\mathcal{T}$. Other vertices may then discover that their cost-to-come would be reduced by updating their parent to this new vertex [1], [4]. While the IS only shrinks when these changes propagate all the way to $v_t$, `GuILD` leverages *any* cost-to-come improvement to adaptively guide densification.

`GuILD` introduces the idea of a *beacon*: a vertex in the search tree that decomposes the original sampling/planning problem into two smaller subproblems (Fig. 3). We define the *Local Subsets* (LS) induced by beacon $b$ to be the union of two prolate hyperspheroids, with foci $(v_s, b)$ and foci $(b, v_t)$. The start-beacon set has transverse axis diameter $g(b)$, only including points that can improve the cost-to-come from $v_s$ to $b$. Given the current shortest subpath to $b$ and its cost-to-come, the beacon-target set only includes points that can extend that subpath and improve the solution cost by setting the transverse axis diameter to $c(\xi) - g(b)$.

`GuILD` adapts $\mathcal{E}_{\text{LS}}$ after each iteration of densification and planning. When $g(b)$ is reduced, the start-beacon set shrinks and the beacon-target set *expands*. Surprisingly, this expansion is actually desirable: if the best solution cost remains the same and the beacon's cost-to-come is reduced, there is a larger path length budget that can be expended between the beacon and target that would still yield a shorter path overall (Fig. 2).

We summarize the `GuILD` framework in Alg. 2, which replaces the starred line of Alg. 1. The BeaconSelector (Alg. 2, Line 4) is a function that chooses a beacon to guide local densification. `GuILD` samples uniformly from $\mathcal{E}_{\text{LS}}$ (Line 6) using the same analytic strategy proposed for $\mathcal{E}_{\text{IS}}$ [39]. `GuILD` must sample from the IS with nonzero probability to retain asymptotic optimality. This is achieved by including $v_s$ in the beacon set.

**Algorithm 3:** Candidate BeaconSelectors

```
    Input: beacon set $\mathcal{B}$
1   Function InformedSet
2       return $v_s$;
3   Function Uniform
4       return $b \sim \mathcal{U}(\mathcal{B})$;
5   Function Greedy
6       return $\arg\max_{b \in \mathcal{B}} w(b)$                ▷ Eq. 4
7   Function Bandit
8       return EXP3($\mathcal{B}$)        ▷ implemented as in [40]
```

In Alg. 3, we present some candidate beacon selectors that we evaluate in Section V. The `InformedSet` beacon selector recovers the behavior of sampling from the IS by choosing $v_s$ as the beacon. `Uniform` uniformly samples from the beacon set $\mathcal{B}$. The `Greedy` beacon selector aims to select the beacon with the maximum possible improvement in path length, while minimizing the measure of the set that needs to be sampled. It takes the ratio of these two quantities:

$$w(b) = \frac{c(\xi) - h(v_s, b) - h(b, v_t)}{\lambda(\mathcal{E}_{\text{LS}})}. \tag{4}$$

Finally, the `Bandit` beacon selector implements the EXP3 adversarial bandit algorithm [40]. The reward function is the fractional improvement in the path length

$$r(b) = \frac{c(\xi_{i-1}) - c(\xi_i)}{c(\xi_{i-1})}. \tag{5}$$

An adversarial bandit algorithm is necessary because this reward function is nonstationary.

### *Properties of Local Subsets*

First, we guarantee that `GuILD` only samples configurations that can improve the current path length by showing that both sets in $\mathcal{E}_{\text{LS}}$ are subsets of the IS (Theorem IV.1). Then, we show that the measure ($n$-volume) of $\mathcal{E}_{\text{LS}}$ is upper-bounded by that of the IS (Theorem IV.2), so choosing an appropriate beacon will allow `GuILD` to densify the space more efficiently.

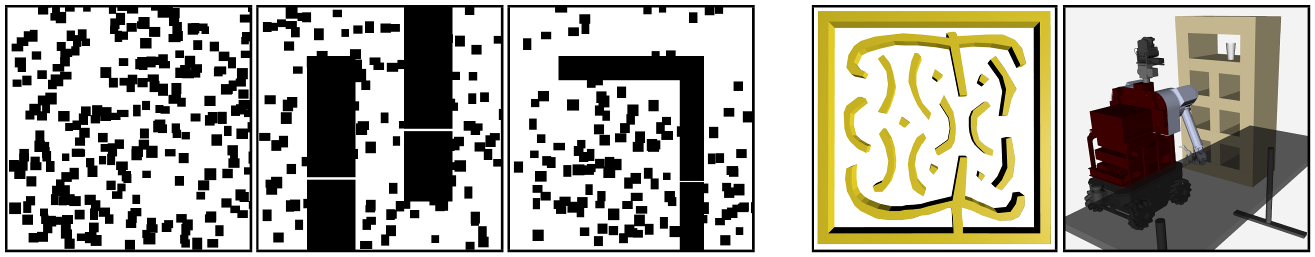
**Theorem IV.1.** *Given the current best solution cost $c(\xi_i)$, a beacon $b \in \mathcal{T}$ previously expanded with cost-to-come $g(b)$, we have that $\mathcal{E}(v_s, b, g(b)),\ \mathcal{E}(b, v_t, c(\xi_i) - g(b)) \subseteq \mathcal{E}_{\text{IS}}$.*

**Theorem IV.2.** *Given the current best solution cost $c(\xi_i)$, a beacon $b \in \mathcal{T}$ previously expanded with cost-to-come $g(b)$, we have $\lambda(\mathcal{E}_{\text{LS}}) \leq \lambda(\mathcal{E}_{\text{IS}})$*

## V. Experiments

We evaluate `GuILD` on an array of planning problems to characterize the proposed beacon selectors (Fig. 4).

In the $\mathbb{R}^2$ environments, the task is to plan from the bottom left corner to the top right corner (`Forest`, `TwoWall`), or the bottom right corner (`Trap`). A forest of obstacles is randomly placed throughout each environment. The easier `Forest` environment only has these random obstacles; as



(a) `Forest` (b) `TwoWall` (c) `Trap` (d) `SE2Maze` (e) `HERB Shelf`

Fig. 4: Evaluation environments.

a result, there are many different homotopy classes that will produce near-optimal paths. `TwoWall` and `Trap` introduce large obstacles with narrow passages that must be crossed to produce near-optimal paths. Discovering these passages typically requires the space to be sampled very densely. At the lower sampling resolutions resulting from fewer initial graph samples, there are many suboptimal homotopy classes that are more easily sampled and discovered. Therefore, initial paths will have high cost and $\lambda(\mathcal{E}_{\text{IS}})$ will be much larger than $\lambda(\mathcal{X})$. These more challenging scenarios highlight the limitations of IS densification.

After characterizing the behavior of `GuILD` with $\mathbb{R}^2$ environments, we consider higher-dimensional planning problems. In `SE2Maze`, a benchmark environment from OMPL [41], the task is to navigate a car through a maze. In `HERB Shelf`, a 7-DOF manipulator [42] is tasked with moving its end-effector to pick an object from a bookshelf.

## *Evaluation Metrics and Hypotheses*

We make three hypotheses that characterize the incremental densification performance of `GuILD`. Sampling from Local Subsets incurs negligible overhead relative to sampling from the Informed Set; both rely on the same analytic sampling primitive for prolate hyperspheroids. To make these results independent of our computing environment, we describe these hypotheses in terms of samples drawn. We compare `GuILD` to `BIT*`, which is equivalent to `GuILD` with the Informed Set BEACONSELECTOR (Alg. 3).

The first metric we consider is the *Sample Efficiency* of incremental densification: how many samples must be drawn before the optimal SBMP algorithm converges to the cost of the optimal path? We determine this minimum cost $c(\xi^*)$ by running with a large timeout (1 min. for $\mathbb{R}^2$ problems, 5 min. for higher-dimensional planning problems).

**H1** *`GuILD` will require fewer samples to converge to the minimum solution cost than the Informed Set.*

Next, to understand the performance of the optimal SBMP algorithm over time, we consider the *Convergence Percentage* and *Normalized Path Cost* as a function of samples. Convergence Percentage is the fraction of trials where the planner had converged to the optimal path cost, with that number of samples. Normalized Path Cost divides the cost of the current best solution by the optimal path cost $\frac{c(\xi)}{c(\xi^*)}$.

**H2** *For a fixed sample budget, `GuILD` will have a higher Convergence Percentage than the Informed Set.*

**H3** *For a fixed sample budget, `GuILD` will have lower Normalized Path Cost than the Informed Set.*

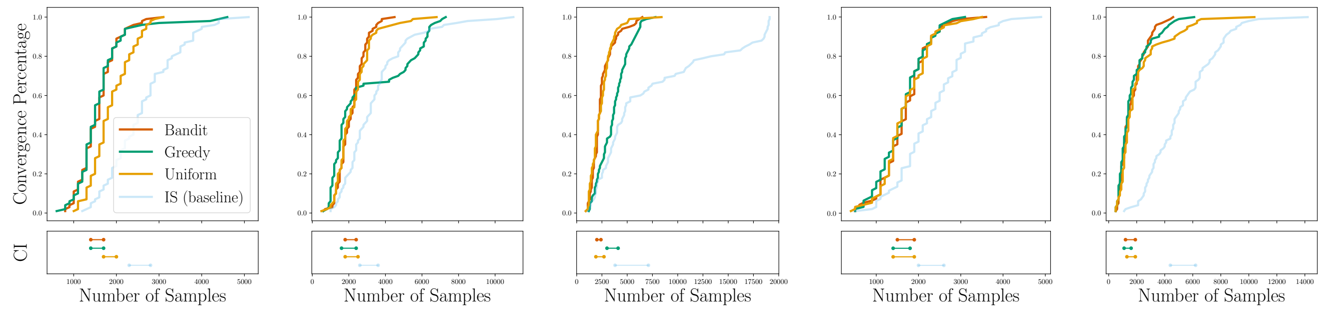
## *Results*

To test these hypotheses, we implement the BEACONSELECTORs described in Section IV. We construct an initial set of beacons $\mathcal{B}$ by sampling states from a low-discrepancy sequence [43]. In our implementation, no new beacons are added to $\mathcal{B}$, and a beacon $b \in \mathcal{B}$ is considered by BEACONSELECTOR only if $b \in \mathcal{E}_{\text{IS}}$. We run 100 random trials for each pair of algorithm and environment.

To understand convergence across the random trials, we plot the Convergence Percentage as a function of the number of samples (Fig. 5). To support **H2**, we would expect the IS baseline curve to remain below the `GuILD` curves. We find this to be the case: on most environments, sampling from the IS results in a lower Convergence Percentage for a fixed sample budget. However, on a third of the `TwoWall` trials, the `Greedy` heuristic causes over-sampling in regions that ultimately do not yield the optimal path.

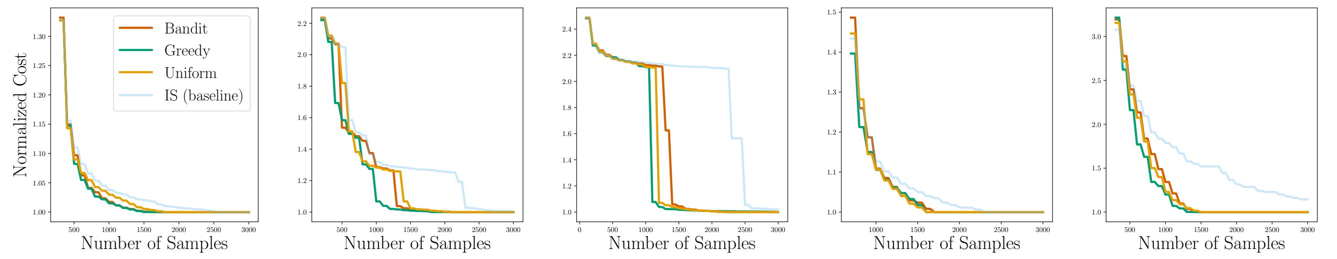
For each environment and instantiation of `GuILD`, Fig. 5 also shows Bonferroni-corrected nonparametric confidence intervals on the median Sample Efficiency; Sample Efficiency is not normally distributed. We find that in almost all instances, `GuILD` focuses sampling more efficiently than sampling from the IS, supporting **H1**. In general, the three `GuILD` selectors achieve comparable Sample Efficiency, suggesting that the key to their success is sampling from Local Subsets. However, the adversarial `Bandit` selector most consistently ranks among the best performing BEACONSELECTORs, while `Uniform` and `Greedy` were each less efficient on one environment.

While **H1** shows that `GuILD` ultimately converges faster to the optimal cost, we would also like to characterize the rate of convergence. For each planning problem, we plot the median Normalized Path Cost across the trials to understand how path length is reduced over time (Fig. 6). Sharp drops in path length (e.g., Fig. 6c) highlight the discrepancy between high-cost homotopies that are easy to sample (navigating around large obstacles) and near-optimal homotopies that are difficult to sample (crossing narrow passages through obstacles). Steady decreases in path length (e.g., Fig. 6a) show planning problems where sampling can more easily discover lower-cost homotopies.



(a) `Forest` (b) `TwoWall` (c) `Trap` (d) `SE2Maze` (e) `HERB Shelf`

Fig. 5: Top: Convergence Percentage across 100 trials per environment. For most environments, sampling from the IS results in a worse Convergence Percentage than `GuILD` for a given sample budget (leftmost curve is better). Bottom: Bonferroni-corrected nonparametric confidence intervals (CI) for the median Sample Efficiency. For all but one `GuILD` instance across environments, the median Sample Efficiency is smaller than that of the IS with 95% confidence. `Bandit GuILD` consistently ranks among the best densification strategies.



(a) `Forest` (b) `TwoWall` (c) `Trap` (d) `SE2Maze` (e) `HERB Shelf`

Fig. 6: Normalized Path Cost, median across 100 trials. Sampling with `GuILD` reaches a near-optimal path length more quickly than sampling from the IS, with significant improvement in all but the easiest planning domain (leftmost curve is better).

**H3** is supported across all environments: sampling with `GuILD` consistently yields a lower cost than sampling with IS. In planning problems with low-cost homotopies, `GuILD` automatically focuses sampling to deliver paths through these narrow passages. With the `Trap` environment, `GuILD` instances identify the narrow passage around 1100-1200 samples and quickly optimize within homotopies crossing the passage. By contrast, the IS baseline requires nearly double the samples to find the passage. The `TwoWall` environment shows a similar trend, although with additional intermediate-cost homotopies that only traverse one of the two narrow passages (Fig. 6b).

The `SE2Maze` and `HERB Shelf` demonstrate the anytime performance of `GuILD` on environments with less distinct homotopy costs. As a result, all sampling schemes steadily improve path cost, similar to `Forest`. However, in `HERB Shelf`, `GuILD` instances have a much steeper improvement, suggesting that `GuILD` may yield more significant sample efficiency improvements in higher dimensions.

## VI. Conclusion

`GuILD` is a new framework for incremental densification with a simple insight: even when the planner fails to discover a shorter path, the search tree still contains valuable information that can immediately improve the densification strategy. `GuILD` selects a *beacon* from the search tree that decomposes the original sampling/planning problem into two smaller subproblems. Improving the cost-to-come for any beacon allows `GuILD` to adaptively shrink and expand the Local Subsets that it samples from. Similar to the Informed Set that `GuILD` builds upon, Local Subsets can be easily (and efficiently) incorporated into any sampling-based optimal motion planning algorithm.

Even simple beacon selectors, such as `Uniform`, can dramatically accelerate the convergence rate relative to the Informed Set densification baseline. We also propose a `Bandit` selector using EXP3, an adversarial bandit algorithm that consistently ranks among the best beacon selectors across all the planning problems we considered. In particular, `GuILD` excels in domains with difficult-to-sample homotopy classes and high-dimensional planning problems.

In this work, we have primarily considered either heuristic beacon selectors (`Greedy`) or beacon selectors that learn from experience online (`Bandit`). Better heuristics for selecting beacons may exist, including ones that build on prior work in identifying and sampling bottleneck points. Embedding these approaches within the `GuILD` framework retains theoretical guarantees of asymptotic optimality, and simplifies the learning problem from generating focused samples to selecting beacons that yield focused samples.

## References

[1] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” *I. J. Robotics Res.*, vol. 30, no. 7, pp. 846–894, 2011.

[2] ——, “Incremental sampling-based algorithms for optimal motion planning,” in *RSS*, 2010.

[3] O. Arslan and P. Tsiotras, "Use of relaxation methods in sampling-based algorithms for optimal motion planning," in *ICRA*, 2013, pp. 2421–2428.

[4] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Batch Informed Trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs," in *ICRA*, 2015, pp. 3067–3074.

[5] M. P. Strub and J. D. Gammell, "Advanced BIT* (ABIT*): Sampling-based planning with advanced graph-search techniques," in *ICRA*, 2020, pp. 130–136.

[6] ——, "Adaptively Informed Trees (AIT*): Fast asymptotically optimal path planning through adaptive heuristics," in *ICRA*, 2020, pp. 3191–3198.

[7] S. Koenig, M. Likhachev, and D. Furcy, "Lifelong planning A*," *Artif. Intell.*, vol. 155, no. 1-2, pp. 93–146, 2004.

[8] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *IROS*, 2014, pp. 2997–3004.

[9] D. Hsu, T. Jiang, J. Reif, and Z. Sun, "The bridge test for sampling narrow passages with probabilistic roadmap planners," in *ICRA*, 2003, pp. 4420–4426.

[10] B. Ichter, J. Harrison, and M. Pavone, "Learning sampling distributions for robot motion planning," in *ICRA*, 2018, pp. 7087–7094.

[11] R. Kumar, A. Mandalika, S. Choudhury, and S. Srinivasa, "LEGO: Leveraging experience in roadmap generation for sampling-based planning," in *IROS*, 2019, pp. 1488–1495.

[12] B. Ichter, E. Schmerling, T.-W. E. Lee, and A. Faust, "Learned critical probabilistic roadmaps for robotic motion planning," in *ICRA*, 2020, pp. 9535–9541.

[13] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

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

[15] J. J. Kuffner and S. M. LaValle, "RRT-Connect: An efficient approach to single-query path planning," in *ICRA*, 2000, pp. 995–1001.

[16] D. Hsu, J.-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," in *ICRA*, 1997, pp. 2719–2726.

[17] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *I. J. Robotics Res.*, vol. 34, no. 7, pp. 883–921, 2015.

[18] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, "MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space," in *ICRA*, 1999, pp. 1024–1031.

[19] C. Holleman and L. E. Kavraki, "A framework for using the workspace medial axis in PRM planners," in *ICRA*, 2000, pp. 1408–1413.

[20] H. Kurniawati and D. Hsu, "Workspace importance sampling for probabilistic roadmap planning," in *IROS*, 2004, pp. 1618–1623.

[21] Y. Yang and O. Brock, "Adapting the sampling distribution in PRM planners based on an approximated medial axis," in *ICRA*, 2004, pp. 4405–4410.

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

[23] J. Denny, E. Greco, S. Thomas, and N. M. Amato, "MARRT: Medial axis biased rapidly-exploring random trees," in *ICRA*, 2014, pp. 90–97.

[24] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, "OBPRM: An obstacle-based PRM for 3D workspaces," in *WAFR*, 1998, pp. 155–168.

[25] V. Boor, M. Overmars, and A. van der Stappen, "The gaussian sampling strategy for probabilistic roadmap planners," in *ICRA*, 1999, pp. 1018–1023.

[26] S. Rodríguez, X. Tang, J.-M. Lien, and N. M. Amato, "An obstacle-based rapidly-exploring random tree," in *ICRA*, 2006, pp. 895–900.

[27] L. Krammer, W. Granzer, and W. Kastner, "A new approach for robot motion planning using rapidly-exploring randomized trees," in *2011 9th IEEE International Conference on Industrial Informatics*. IEEE, 2011, pp. 263–268.

[28] S. Kiesel, E. Burns, and W. Ruml, "Abstraction-guided sampling for motion planning," in *International Symposium on Combinatorial Search*, vol. 3, no. 1, 2012.

[29] C. Chamzas, A. Shrivastava, and L. E. Kavraki, "Using local experiences for global motion planning," in *ICRA*, 2019, pp. 8606–8612.

[30] C. Chamzas, Z. Kingston, C. Quintero-Peña, A. Shrivastava, and L. E. Kavraki, "Learning sampling distributions using local 3d workspace decompositions for motion planning in high dimensions," in *ICRA*, 2021, pp. 1283–1289.

[31] A. Khan, A. Ribeiro, V. Kumar, and A. Francis, "Graph neural networks for motion planning," *CoRR*, vol. abs/2006.06248, 2020.

[32] J. Denny and N. M. Amato, "Toggle PRM: A coordinated mapping of C-free and C-obstacle in arbitrary dimension," in *WAFR*, 2012, pp. 297–312.

[33] L. Jaillet, J. Cortés, and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *IEEE Trans. Robotics*, vol. 26, no. 4, pp. 635–646, 2010.

[34] F. Hauer and P. Tsiotras, "Deformable rapidly-exploring random trees," in *RSS*, 2017.

[35] J. M. Phillips, N. Bedrossian, and L. E. Kavraki, "Guided Expansive Spaces Trees: A search strategy for motion- and cost-constrained state spaces," in *ICRA*, 2004, pp. 3968–3973.

[36] O. Arslan and P. Tsiotras, "Dynamic programming guided exploration for sampling-based motion planning algorithms," in *ICRA*, 2015, pp. 4819–4826.

[37] ——, "Machine learning guided exploration for sampling-based motion planning algorithms," in *IROS*, 2015, pp. 2646–2652.

[38] S. S. Joshi and P. Tsiotras, "Relevant region exploration on general cost-maps for sampling-based motion planning," in *IROS*, 2020, pp. 6689–6695.

[39] J. D. Gammell, T. D. Barfoot, and S. S. Srinivasa, "Batch Informed Trees (BIT*): Informed asymptotically optimal anytime search," *I. J. Robotics Res.*, vol. 39, no. 5, pp. 543–567, 2020.

[40] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The nonstochastic multiarmed bandit problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002.

[41] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[42] S. S. Srinivasa, D. Ferguson, C. J. Helfrich, D. Berenson, A. Collet, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. V. Weghe, "HERB: a home exploring robotic butler," *Auton. Robots*, vol. 28, no. 1, pp. 5–20, 2010.

[43] J. H. Halton and G. B. Smith, "Algorithm 247: Radical-inverse quasi-random point sequence," *Comm. of the ACM*, vol. 7, no. 12, pp. 701–702, 1964.