2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Cooperative Informed Tree (CoIT*): Cooperative Bi-Directional Multi-Resolution Motion Planning with Adaptive Edge Screening

Xiao Tan[1], Yaonan Wang[1], Renjie Ding[1], Min Liu[1], Zhe Zhang[2], and Xiaoqian Yu[1]

***Abstract*— In informed search-based path planning, heuristic functions that incorporate problem knowledge are essential for guiding the search and improving efficiency. The accuracy and computational cost of these heuristics are therefore critical to performance. However, accuracy and computational efficiency are often contradictory, making it difficult to select an appropriate heuristic for a given problem. In this paper, we present CoIT* (Cooperative Informed Tree*), an almost-asymptotically optimal asymmetric bi-directional planning algorithm designed to address these challenges. CoIT* introduces a multi-resolution and multi-heuristic queue cooperation mechanism between forward and reverse searches: the forward search interacts with the reverse search to provide cooperative information exchange, which enhances both local and global edge screening. This cooperation improves the accuracy of the reverse search, while multi-resolution exploration enables lazy edge validation in the forward search, thereby reducing planning time. We validate CoIT* on high-dimensional benchmark problems as well as simulated and real surgical robot planning tasks. Experimental results demonstrate that CoIT* achieves higher accuracy and significantly lower planning time compared with state-of-the-art planners.**

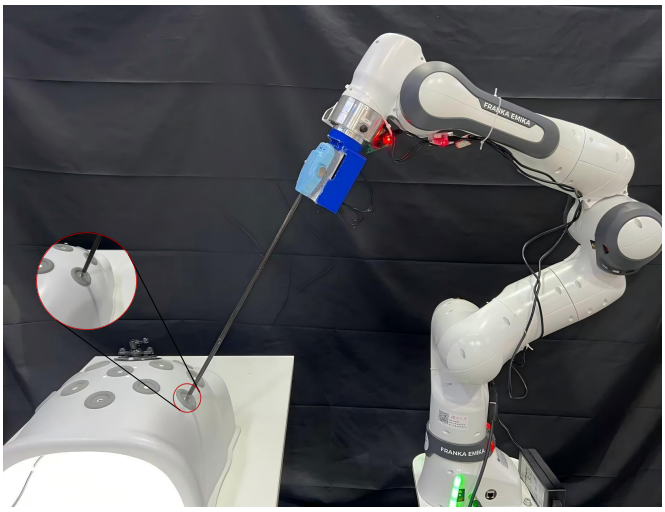


Fig. 1: Robotic instrument insertion into the body under an abdominal surgical-instrument environment.

I. Introduction

Path-planning algorithms aim to compute feasible states from start to goal and to find collision-free, safe trajectories. Graph-search methods such as A* [1] discretize the environment and search in configuration space (*$\mathcal{C}$-space*). However, finer discretization leads to a state-space explosion. To improve upon graph search and alleviate discretization issues, MRA* [2] performs fast exploration at coarse resolution combined with trajectory refinement at fine resolution; MHA* [3] accelerates search by leveraging multiple heuristics; AMHA* [4] provides improvements over MHA* [3]; and AMRA* [5] combines multiple heuristics with multi-resolution search, achieving complete planning via heuristic information sharing and cooperative multi-resolution reasoning.

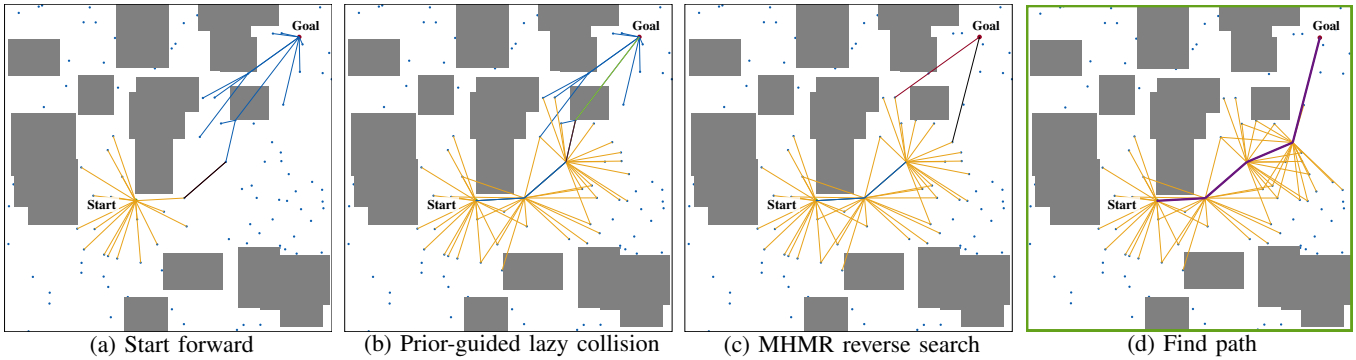
Nevertheless, graph-based planning still suffers from the curse of dimensionality [6], [7]. To address this, sampling-based algorithms [8] were proposed, including the Probabilistic Roadmap (PRM) [9] and Rapidly-Exploring Random Trees (RRT) [10]. Their unordered growth and the heavy cost of collision checking can severely impact performance. This has been mitigated to some extent by introducing heuristics and lazy collision checking into sampling-based planning. RRT-Connect [11] grows two trees incrementally and uses a connection heuristic to expand rapidly, but it is not asymptotically optimal. Informed RRT* [12] injects an ellipsoidal informed set as prior knowledge into RRT* [13], accelerating convergence while preserving asymptotic optimality. Motion Planning using Lower Bounds [14] is an anytime variant of FMT* [15] that reduces the number of expanded nodes and collision checks via two Dijkstra runs, though local planning remains time-consuming.

Later, Batch Informed Trees (BIT*) [16], [17] combined the ellipsoidal informed set with an admissible cost heuristic to form an implicit random geometric graph (RGG) with gradually increasing density [18]. However, its heuristic is fixed; for some problems such a heuristic may not exist or be effective. To speed up BIT*'s initial search, Advanced BIT* [19] introduces graph-search techniques of inflation and truncation, but at the cost of increasing the number of edge evaluations. Adaptively Informed Trees (AIT*) [7], [20] and Effort Informed Trees (EIT*) [7] employ asymmetric bidirectional search, where reverse-side information provides heuristic guidance to the forward search; however, reverse-side edge screening is limited, and the information provided for forward feasible edges is under-exploited. Direction Informed Trees (DIT*) [21] applies a directional filter within the local region to select feasible edges, but the induced directional bias can trap the search in local optima. Therefore, properly exploiting cooperative information between forward and reverse searches can both accelerate edge screening and more fully leverage heuristic guidance to speed up the planning process.

In parallel, lazy edge-evaluation methods, such as LazySP and its extensions, formalize deferred edge validation and adaptive checking policies to reduce the cost of expensive edge evaluations [22], [23], [24]. These methods show that interleaving search and selective edge validation can substantially improve planning efficiency, but they are not designed to exploit cooperative forward–reverse heuristic interactions

[1]X. Tan, Y. Wang, R. Ding, M. Liu, and X. Yu are with the School of Artificial Intelligence and Robotics, the National Engineering Research Center of Robot Visual Perception and Control Technology, Hunan University, Changsha, Hunan 410082, China. (Corresponding author: Yaonan Wang.) (e-mail: xiaotan9426@hnu.edu.cn)

[2]Z. Zhang is with Guangxi University, Nanning, Guangxi, China.



(a) Start forward (b) Prior-guided lazy collision (c) MHMR reverse search (d) Find path

Fig. 2: Four snapshots of the CoIT* planner for shortest-path planning: (a) initialize a random geometric graph (RGG) and run a multi-resolution reverse search from the goal with an admissible cost heuristic, then start the forward search; (b) prior-guided lazy collision checking: before validating the current forward edge (purple), first test the incident reverse edge (green) at a higher resolution; (c) after a forward collision, the reverse search uses previously validated feasible edges to guide a multi-heuristic, multi-resolution expansion (red: best edge under an admissible heuristic; black: exploratory best under an inadmissible heuristic); (d) with reverse guidance, the forward search accelerates and finds a path.

in informed tree search. For kinodynamic planning, informed sampling has also been generalized beyond fixed heuristics, e.g., via MCMC-based strategies that improve sample efficiency in complex dynamics-constrained settings [25]. More recently, bidirectional lazy heuristic search has been integrated into batch-wise sampling-based motion planning (e.g., BLIT*) to accelerate anytime convergence through lazy bidirectional expansion [26]. Motivated by these directions, CoIT* unifies cooperative bidirectional heuristic guidance with adaptive multi-resolution lazy edge screening. In particular, our Adaptive Edge Evaluation Screening (AEES) can be viewed as a multi-resolution extension of LazySP-style deferred evaluation, while cooperative forward–reverse queue interactions further prioritize when and how edges are screened during planning.

CoIT* (Cooperative Informed Tree*) is a bi-directional, asymmetric, cooperative path-planning framework. On the *reverse* side (from the goal), it employs a *multi-heuristic*, *multi-resolution* strategy to improve the quality and speed of the reverse search. On the *forward* side (from the start), the search expands under these priors and performs *prior-guided lazy collision checking* [27]: before executing the expensive full-resolution test, it first conducts a high-resolution pre-check on the reverse incident edge of the current best forward edge, thereby deferring the full test and significantly reducing futile exploration and collision-checking overhead.

As an efficient planner, CoIT* simultaneously enhances the accuracy of the reverse search in bi-directional planning and improves the efficiency of forward validation. The method is validated in both surgical-robot simulation and real hardware experiments (Fig. 1), and in high-dimensional environments it achieves significant improvements in runtime and path cost compared with state-of-the-art planners such as EIT* and AIT*.

The contributions of this work are summarized as follows:

- *Multi-resolution cooperative framework:* a multi-heuristic, multi-resolution reverse search supplies accurate and efficient priors to the forward search; the forward search, in turn, leverages high-resolution priors on reverse edges, and the forward tree's region of expansion feeds back to steer the reverse search.
- *Local and global resolution:* CoIT* adaptively refines both local and global resolutions to accelerate edge screening and prune a large number of infeasible edges.
- *Practical applicability and effectiveness:* CoIT*'s planning-efficiency gains are validated in simulation and real surgical-robot tasks, as well as in high-dimensional planning problems.

II. BACKGROUND

Before presenting CoIT*, we first formalize the optimal motion-planning problem and the notation used throughout.

A. Problem Definition

We define the motion-planning problem within the framework of sampling-based methods [13]. Let the state space be $X \subseteq \mathbb{R}^n$. Let $X_{\text{obs}} \subset X$ denote obstacle (invalid) states, and let $X_{\text{free}} = cl\big(X \setminus X_{\text{obs}}\big)$ be the valid, collision-free states, where $cl(\cdot)$ denotes set closure. Given an initial state $x_{\text{start}} \in X_{\text{free}}$ and a goal set $X_{\text{goal}} \subset X_{\text{free}}$, we denote a goal state by $x_{\text{goal}} \in X_{\text{goal}}$ (e.g., sampled from X_{goal} when initializing the reverse search). A path is a continuous map $\sigma : [0,1] \to X_{\text{free}}$ such that $\sigma(0) = x_{\text{start}}$ and $\sigma(1) \in X_{\text{goal}}$. Let Σ denote the set of all nontrivial feasible paths, and let $c : \Sigma \to \mathbb{R}_{\geq 0}$ be a nonnegative cost functional. The optimal planning problem is

$$\sigma^* = \arg\min_{\sigma \in \Sigma} \{c(\sigma) | \sigma(0) = \mathbf{x}_{\text{start}}, \sigma(1) \in \mathbf{x}_{\text{goal}}, \forall t \in [0,1], \sigma(t) \in X_{\text{free}}\}. \tag{1}$$

and the optimal cost is $c^* = c(\sigma^*)$.

When planning is performed over a discrete set of samples $X_{\text{samples}} \subset X$ drawn i.i.d. from a sampling measure μ (e.g., $x_i \overset{\text{i.i.d.}}{\sim} \mathcal{U}(X_{\text{free}})$ and $X_{\text{samples}} = \{x_i\}_{i=1}^N$), the induced roadmap can be modeled as a (possibly implicit) random geometric graph (RGG) [18], with edges determined by a local planner or a transition rule.

B. Notation

CoIT* operates in a planning space $X \subseteq \mathbb{R}^n$ with a start state $\mathbf{x}_{\text{start}} \in X$, a goal set $X_{\text{goal}} \subset X$, and a set of sampled states $X_{\text{sampled}} \subset X$ (Fig. 2). The search is bidirectional, maintaining a forward tree $\mathcal{F} = (V_{\mathcal{F}}, E_{\mathcal{F}})$ and a reverse tree $\mathcal{R} = (V_{\mathcal{R}}, E_{\mathcal{R}})$. Vertices $V_{\mathcal{F}}$ and $V_{\mathcal{R}}$ correspond to valid states.

An edge is denoted by $e = (\mathbf{x}_s, \mathbf{x}_t)$ with source $\mathbf{x}_s$ and target $\mathbf{x}_t$. Forward edges $E_{\mathcal{F}} \subseteq V_{\mathcal{F}} \times V_{\mathcal{F}}$ denote edges that are

accepted as feasible after full-resolution validation. Reverse edges $E_{\mathcal{R}} \subseteq V_{\mathcal{R}} \times V_{\mathcal{R}}$ provide heuristic guidance and may be validated sparsely at non-full resolutions.

We use $\mathcal{Q}_F$ for the forward edge queue, $\mathcal{Q}_R$ for the reverse queue, and $\mathcal{Q}_{\text{PriR}}$ for a reverse priority queue that leverages an inadmissible expansion heuristic $\bar{se}$ to accelerate forward–reverse connection. When needed, an admissible estimate $\hat{f} : X \to [0, \infty)$ defines an informed set $X_{\hat{f}}$.

Multi-resolution collision checking: CoIT* employs three resolutions: a global working resolution r, a higher (but still inexpensive) pre-check resolution r_{high}, and a full resolution r_{full}. Each state $\mathbf{x}$ maintains a *local resolution level* $\text{localRes}(\mathbf{x})$, which is refined online; the refinement rule and triggering conditions are described in Section III-B.

The reverse search performs (i) sparse collision checking at the global resolution r to cheaply screen reverse edges, and (ii) higher-resolution checking at r_{high} to refine local resolution levels and strengthen guidance. The forward search performs prior-guided lazy collision checking using r_{high} as a pre-check, and ultimately validates candidate forward edges at r_{full} to decide feasibility.

Here, resolution refers to the sampling density along an edge rather than graph discretization. For example, one may use an odd sample count $N(\bar{r}) = 2k+1$ to support midpoint refinement.

III. Cooperative Informed Trees (CoIT*)

This section is organized as follows. We first present the multi-resolution cooperative framework and show how it enables fast bi-directional search. We then introduce local and global resolution mechanisms that leverage collision-checking feedback to strengthen edge screening. Finally, we prove that CoIT* guarantees probabilistic completeness and asymptotic optimality.

A. Multi-Resolution Cooperative Framework

R-Nearest & K-Nearest neighbor connections, Given $N = |X_{\text{sampled}}|$ samples, we consider two standard connection rules on the roadmap: (i) the *r-nearest* rule connects a state x to all other states within a radius $r(N)$; and (ii) the *k-nearest* rule connects x to its $k(N)$ closest states in Euclidean distance. A sufficient choice (with a safety factor $\gamma \geq 1$) is

$$r(N) \;\geq\; \eta_r \left(2\left(1 + \frac{1}{n}\right) \frac{\lambda\left(X_{\hat{f}}\right)}{\zeta_n} \frac{\log N}{N} \right)^{\frac{1}{n}}, \qquad (2)$$

where $\eta_r > 0$ is a normalization/tuning constant, n is the state-space dimension, $\lambda(\cdot)$ is the Lebesgue measure, $X_{\hat{f}}$ is the informed set (one may replace it with X_{free} for global planning), and $\zeta_n = \lambda(B_{1,n})$ is the volume of the n-dimensional unit ball $B_{1,n}$.

For the k-nearest rule, we set:

$$k(N) \;\geq\; \eta_k \left(1 + \frac{1}{n}\right) \log N, \qquad (3)$$

where $\eta_k > 0$ is a tuning parameter (absorbing constant factors such as the natural-base e if desired). The choices in (2)–(3) capture the typical $\left(\frac{\log N}{N}\right)^{1/n}$ and $\log N$ scalings used for sparse, well-connected RGGs as N grows.

Here we introduce a new heuristic (an inadmissible expand heuristic). For x_t in Q_F, it stores the minimum cost from the start to that state (see the x_t indicated by the yellow edge in (Fig. 2), analogous to LPA*. It is updated as the forward search proceeds (Alg. 1, line 23):

$$\bar{g}_{\text{fe}}(x_t) = \min \{\, g(x_s) + \widehat{c}(x_s, x_t) \,\}. \qquad (4)$$

In Q_R, entries are maintained in an A*-like manner with an admissible heuristic, so the reverse search remains goal-directed toward X_{goal}. However, forward-side truncated edges produced earlier still provide useful guidance: they encode feasible prefixes from the start to their targets and, because truncation was triggered by local collisions, they implicitly capture nearby obstacle information. Leveraging this, a multi-heuristic, multi-resolution strategy can more effectively filter neighboring edges and quickly form connections to the forward tree. To realize this, we introduce the

Algorithm 1: Cooperative Informed Trees (CoIT*)

1 $c_{\text{current}} \leftarrow \infty$
2 $\xi_i \leftarrow$ `updateInitialFactor()`
3 $r \leftarrow$ `setInitialResolution()`
4 $r_{\text{high}} \leftarrow$ `setHighResolution()`
5 *Set the local resolution of* $\mathbf{x}_{start}$ *and* $\mathbf{x}_{goal}$ *to* r
6 $X_{\text{sampled}} \leftarrow X_{\text{goal}} \cup \{\mathbf{x}_{\text{start}}\}$
7 $V_{\mathcal{F}}, E_{\mathcal{F}}, Q_F \leftarrow$ `expand`$(\mathbf{x}_{\text{start}})$
8 $V_{\mathcal{R}}, V_{\mathcal{R},\text{closed}}, E_{\mathcal{R}}, (Q_R, Q_{\text{PriR}}) \leftarrow$ `expand`(X_{goal})
9 **repeat**
10 **if**
Reverse queue has better edge priority than forward queue ***or*** *target of optimal edge in forward queue not closed* **then**
11 $((\mathbf{x}_s, \mathbf{x}_t), r_i) \leftarrow$ `getBestReverseEdge`(Q_R, Q_{PriR})
12 $V_{\mathcal{R},\text{closed}} \leftarrow V_{\mathcal{R},\text{closed}} \cup \{\mathbf{x}_s\}$
13 $r_{\text{current}} \leftarrow \max(r_i, \text{localRes}(\mathbf{x}_s))$
14 **if** `mResNoColl`$((\mathbf{x}_s, \mathbf{x}_t), r_{current})$ **then**
15 *Update cost and reverse tree structures*
16 **else if**
$\min_{(\mathbf{x}_s, \mathbf{x}_t) \in \mathcal{Q}_{\mathcal{F}}} \left(\widehat{g}(\mathbf{x}_t) + \widehat{c}(\mathbf{x}_s, \mathbf{x}_t) + \widehat{h}(\mathbf{x}_s) \right) < c_{current}$ **then**
17 $(\mathbf{x}_s, \mathbf{x}_t) \leftarrow$ `getBestForwardEdge`(Q_F)
18 $Q_F \leftarrow Q_F \setminus \{(\mathbf{x}_s, \mathbf{x}_t)\}$
19 **if** $(\mathbf{x}_s, \mathbf{x}_t) \in E_{\mathcal{F}}$ **then**
20 *Update forward tree structures*
21 **else if** $(\mathbf{x}_s, \mathbf{x}_t)$ *improves the forward tree* **then**
22 **if** `mResNoColl`$((\text{parent}_R(\mathbf{x}_t), \mathbf{x}_t), r_{high})$ ***and*** `mResNoColl`$((\mathbf{x}_s, \mathbf{x}_t), r_{full})$ **then**
23 *Update cost and forward tree structures*
24 **if** $\min\{g_{\mathcal{F}}(\mathbf{x}_{\text{goal}})\} < c_{current}$ **then**
25 $\xi_i \leftarrow$ `updateInitialFactor()`
26 **else if** $(\mathbf{x}_t, \mathbf{x}_s) \in E_{\mathcal{R}}$ **then**
27 $r \leftarrow$ `updateResolution()`
28 $(Q_R, Q_{\text{PriR}}) \leftarrow$ `expand`(X_{goal})
29 **else**
30 `prune`(X_{sampled})
31 $X_{\text{sampled}} \xleftarrow{+}$ `sample`(m, c_{current})
32 $r \leftarrow$ `resetResolution()`
33 *Set the local resolution of new samples to* r
34 $Q_F \leftarrow$ `expand`$(\mathbf{x}_{\text{start}})$
35 $(Q_R, Q_{\text{PriR}}) \leftarrow$ `expand`(X_{goal})
36 **until** *planner termination condition*

Algorithm 2: CoIT* - select reverse edge and resolution

Input : reverse queue Q_R, reverse Priority Queue Q_{PriR}
Output: edge $(\mathbf{x}_s, \mathbf{x}_t)$, collision resolution r_i

Function `getBestReverseEdge` (Q_R, Q_{PriR})
  if $\xi_i \neq \infty$ ***and*** $Q_{\mathrm{PriR}} \neq \emptyset$ **then**
    $(\mathbf{x}_s^{\mathrm{peek}}, \mathbf{x}_t^{\mathrm{peek}}) \leftarrow \mathrm{peek}(Q_R)$
    if $\min_{(\mathbf{x}_s,\mathbf{x}_t)\in Q_{\mathrm{PriR}}} \{\mathrm{key}_{\mathrm{PriR}}^{\mathrm{CoIT}^*}(\mathbf{x}_s,\mathbf{x}_t)\} \leq \bar{se}(\mathbf{x}_s^{\mathrm{peek}}, \mathbf{x}_t^{\mathrm{peek}})$ **then**
      $(\mathbf{x}_s, \mathbf{x}_t) \leftarrow \arg\min_{(\mathbf{x}_s,\mathbf{x}_t)\in Q_{\mathrm{PriR}}} \{\mathrm{key}_{\mathrm{PriR}}^{\mathrm{CoIT}^*}(\mathbf{x}_s,\mathbf{x}_t)\}$;
      $r_i \leftarrow r_{\mathrm{high}}$
      $Q_{\mathrm{PriR}} \leftarrow Q_{\mathrm{PriR}} \setminus \{(\mathbf{x}_s, \mathbf{x}_t)\}$
      return $((\mathbf{x}_s, \mathbf{x}_t), r_i)$
  $(\mathbf{x}_s, \mathbf{x}_t) \leftarrow \arg\min_{(\mathbf{x}_s,\mathbf{x}_t)\in Q_R} \{\mathrm{key}_{\mathcal{R}}^{\mathrm{CoIT}^*}(\mathbf{x}_s,\mathbf{x}_t)\}$
  $r_i \leftarrow r$
  $Q_R \leftarrow Q_R \setminus \{(\mathbf{x}_s, \mathbf{x}_t)\}$
  return $((\mathbf{x}_s, \mathbf{x}_t), r_i)$

priority reverse queue Q_{PriR} together with a high-resolution pre-check.

$$\mathrm{key}_{\mathrm{PriR}}^{\mathrm{CoIT}^*}(\mathbf{x}_s,\mathbf{x}_t) := \begin{cases} \bar{h}[\mathbf{x}_s] + \widehat{c}(\mathbf{x}_s,\mathbf{x}_t) + \bar{g}_{\mathrm{fe}}(\mathbf{x}_t), \\ \widehat{h}[\mathbf{x}_s] + \widehat{c}(\mathbf{x}_s,\mathbf{x}_t) + \widehat{g}(\mathbf{x}_t), \\ \bar{e}[\mathbf{x}_s] + \bar{e}(\mathbf{x}_s,\mathbf{x}_t) + \bar{d}(\mathbf{x}_t). \end{cases} \tag{5}$$

where $\bar{g}[\cdot]$, $\widehat{g}[\cdot]$ and $\bar{d}[\cdot]$ denote the inadmissible expand cost, the admissible prior cost, and the inadmissible prior effort for a path from the target state $\mathbf{x}_t$ to the start. This key represents, respectively: (i) the potential path that traverses the edge and connects to the forward expansion region, (ii) the overall potential solution cost, and (iii) the overall potential computational effort.

At initialization, the forward side adopts the *Anytime Explicit Estimation Search* (AEES) [28] key as its heuristic, while the reverse side is anchored by an *admissible* key on Q_R; in addition, the priority reverse queue Q_{PriR} provides an *expand* (inadmissible) key as an auxiliary cue. The scheduler is governed by the initial factor ξ_i: when $\xi_i = \infty$ (time-first regime), we run reverse search alone until it reaches the forward-reachable region, at which point the forward search is activated. When $\xi_i < \infty$ (cost-convergence regime), the forward search is enabled only if the forward AEES key is smaller than the reverse anchor key, i.e.,

$$\min_{(\mathbf{x}_s,\mathbf{x}_t)\in Q_F} \mathrm{key}_{\mathcal{F}}^{\mathrm{AEES}}(\mathbf{x}_s,\mathbf{x}_t) < \min_{(\mathbf{x}_s,\mathbf{x}_t)\in Q_R} \mathrm{key}_{\mathcal{R}}^{\mathrm{anchor}}(\mathbf{x}_s,\mathbf{x}_t), \tag{6}$$

which corresponds to Alg. 1, line 10. Correspondingly, as in EIT*, the forward heuristic is effort-based and the reverse heuristic is admissible; see [7].

In Alg. 1, line 11 selects the reverse edge and its matching resolution. If the minimum key in the priority reverse queue is smaller than $\bar{se}(\mathbf{x}_s^{\mathrm{peek}}, \mathbf{x}_t^{\mathrm{peek}})$ (Alg. 2, line 4):

$$\bar{se}\left(\mathbf{x}_s^{\mathrm{peek}}, \mathbf{x}_t^{\mathrm{peek}}\right) := \bar{h}\left[\mathbf{x}_s^{\mathrm{peek}}\right] + \widehat{c}\left(\mathbf{x}_s^{\mathrm{peek}}, \mathbf{x}_t^{\mathrm{peek}}\right) + \bar{g}_{\mathrm{fe}}\left(\mathbf{x}_t^{\mathrm{peek}}\right). \tag{7}$$

Then we pop the selected edge from the priority reverse queue and set the checking resolution to the high level r_{high}, which indicates that the reverse search has approached either a forward truncation region or the neighborhood of the start; hence a high resolution is adopted to strengthen the screening of adjacent edges (Fig. 2(c)).

Notably, when the selected reverse edge comes from Q_{PriR}, after multi-resolution checking, even if it is collision-free we do *not* expand its endpoint; this keeps Q_R strictly ordered. Conversely, if the popped edge from Q_{PriR} fails the multi-resolution check (i.e., a collision is detected), we insert that edge (both directions) into the `Blacklist`. There is no need to touch Q_R: when Q_R later encounters the same edge, the blacklist is consulted first and the edge is immediately discarded. These two cases avoid reordering side effects and preserve the A*-like ordering of Q_R. As shown in Alg. 1, line 28 and Alg. 3, line 14, the expansion of Q_R then augments Q_{PriR}, ensuring that the overall search remains anchored by an admissible heuristic.

When the best reverse edge incurs no collision at high resolution and reaches the forward-reachable region, the forward search starts (Fig. 2(a)). Before executing the expensive full-resolution collision check on a forward candidate edge $(\mathbf{x}_s, \mathbf{x}_t)$, CoIT* first consults the outcome of the reverse incident edge to decide whether to defer the full check, thereby avoiding unnecessary r_{full} evaluations and improving efficiency (Alg. 1, line 22). If the reverse incident edge $\left(\mathtt{parent}_R(\mathbf{x}_t), \mathbf{x}_t\right)$ fails the high-resolution pre-check at r_{high}, the algorithm restarts the reverse search (Fig. 2(b)). Edges that pass at r_{full} are inserted into the `Whitelist`, whereas edges that pass at r_{high} are cached in the `PreChecklist` for reuse. This mechanism enables the forward expansion to avoid many futile checks under the guidance of reverse-side priors (Fig. 2(b)) while preserving correctness: every edge included in the final path is ultimately validated at r_{full}.

CoIT* introduces lightweight bookkeeping beyond AIT*/EIT*, including an additional reverse priority queue Q_{PriR}, resolution-aware caches (`Whitelist`, `Blacklist`, `PreChecklist`), and per-edge records (e.g., r_{already}) to avoid redundant checks. These operations are mainly queue updates and hash-set lookups and are typically dominated by collision checking in high-dimensional planning. In practice, this overhead is offset by reducing expensive full-resolution evaluations at r_{full} via reverse-side pre-checking at r_{high} and reuse of partial validation results.

B. Local and Global Resolution

Compared with AIT*, where the reverse search does not perform collision checking, and EIT*, where the reverse search adopts adaptive sparse collision checking, CoIT* combines multi-resolution screening and full-resolution validation to form a unified global–local adaptive resolution mechanism. This enables the reverse search to balance exploration in both sparse and cluttered regions, thereby accelerating edge screening.

Different from EIT*, which relies on forward truncation to increase the global resolution, CoIT* also supports local adaptive refinement during reverse high-resolution collision checking at r_{high} (Alg. 1, line 13; Alg. 3, line 20). Each edge $e = (\mathbf{x}_s, \mathbf{x}_t)$ is discretized via hierarchical midpoint subdivision, and the local resolution of a vertex $\mathbf{x}$ is denoted by $\mathtt{localRes}(\mathbf{x})$. We use r_{already} as a bookkeeping variable to exclude collision-checking work that has already been

Algorithm 3: CoIT* - mResNoColl$((\mathbf{x}_s, \mathbf{x}_t), \bar{r})$ for multi-resolution collision checking

Input : edge $(\mathbf{x}_s, \mathbf{x}_t)$, collision resolution $\bar{r}$
Output: bool edgeIsValid

1 **if** $(\mathbf{x}_s, \mathbf{x}_t) \in$ *Whitelist* **then**
2 **return true**
3 **if** $(\mathbf{x}_s, \mathbf{x}_t) \in$ *Blacklist* **then**
4 **return false**
5 **if** $\bar{r} \leq r_{high}$ ***and*** $(\mathbf{x}_s, \mathbf{x}_t) \in$ *PreChecklist* **then**
6 **return true**
7 $r_{\text{full}} \leftarrow$ fullCheck$((\mathbf{x}_s, \mathbf{x}_t))$ ▷ full-resolution Check
8 $r_{\text{already}} \leftarrow$ checked$((\mathbf{x}_s, \mathbf{x}_t))$ ▷ performed Check
9 indices $\leftarrow \{(1, \bar{r})\}$; $currentCheck = 1$
10 **while** *indices* $\neq \emptyset$ **do**
11 $(l, k) \leftarrow$ pop(indices);
12 **if** $currentCheck > r_{already}$ **then**
13 **if** ***no*** pointValid (midPoint$((l, k))$) **then**
14 Blacklist $\xleftarrow{+} \{(\mathbf{x}_s, \mathbf{x}_t), (\mathbf{x}_t, \mathbf{x}_s)\}$
15 **if** $\bar{r} = r_{full}$ ***and*** $currentCheck >$ (localRes$(\mathbf{x}_t) - 1)/2$ **then**
16 localRes$(\mathbf{x}_t) \leftarrow 2 \cdot$ localRes$(\mathbf{x}_t) + 1$
17 **else if** $\bar{r} \neq r_{full}$ ***and*** $currentCheck >$ (localRes$(\mathbf{x}_s) - 1)/2$ **then**
18 **if** $currentCheck \geq r$ **then**
19 $r \leftarrow$ updateResolution()
20 localRes$(\mathbf{x}_s) \leftarrow 2 \cdot$ localRes$(\mathbf{x}_s) + 1$
21 **return false**
22 indices $\xleftarrow{+}$ updateIndices$((l, k))$
23 $currentCheck \leftarrow currentCheck + 1$
24 checked$((\mathbf{x}_s, \mathbf{x}_t)) \leftarrow currentCheck - 1$
25 **if** $\bar{r} = r_{full}$ **then**
26 Whitelist $\xleftarrow{+} \{(\mathbf{x}_s, \mathbf{x}_t), (\mathbf{x}_t, \mathbf{x}_s)\}$
27 **if** $\bar{r} = r_{high}$ **then**
28 PreChecklist $\xleftarrow{+} \{(\mathbf{x}_s, \mathbf{x}_t), (\mathbf{x}_t, \mathbf{x}_s)\}$
29 **return true**

TABLE I: Adaptive refinement examples

Collision index	Sample type	Refinement
2	Old (in res-3)	None
5	New (in res-7)	localRes$(\mathbf{x}) \leftarrow 15$

performed on e: once e has been checked up to level r_{already}, we do not re-evaluate it at any coarser or equal resolution, thereby avoiding redundant computation.

When a collision is detected, CoIT* distinguishes two cases.

Old sample (present at a coarser resolution). The collision would also have been observed at the previous level, so refinement is unnecessary.

New sample (introduced at the current resolution). This indicates that the previous resolution was insufficient. In this case, the local resolution is refined according to

$$\texttt{localRes}(\mathbf{x}_s) \leftarrow 2 \cdot \texttt{localRes}(\mathbf{x}_s) + 1, \tag{8}$$

or similarly for $\mathbf{x}_t$ depending on the edge direction.

Formally, refinement is triggered when

$$\texttt{currentCheck} > \frac{\texttt{localRes}(\mathbf{x}) - 1}{2}, \tag{9}$$

which means the collision occurred at a newly introduced midpoint rather than an old sample. In addition, as shown in Alg. 3, line 15, forward collision checking at full resolution r_{full} can also trigger refinement of the reverse local resolution. This bidirectional coupling ensures that local refinements are consistently synchronized between forward and reverse searches.

Table I provides an example with $\texttt{localRes}(\mathbf{x}) = 7$, illustrating that old-sample collisions do not refine $\texttt{localRes}(\mathbf{x})$, whereas new-sample collisions trigger upscaling. Here, res-k denotes an edge-sampling level (i.e., the sampling density along the edge). Beyond local adaptivity, CoIT* also incorporates a global resolution mechanism. Similar to EIT*, the global resolution is increased when a forward edge is truncated (Alg. 1, line 27). Moreover, CoIT* extends this principle to the reverse search: when a reverse collision is detected and the current check index already exceeds the global resolution, the global resolution is also increased (Alg. 3, line 19). This symmetric rule ensures that both forward and reverse searches consistently trigger global refinement.

C. Probabilistic Completeness and Asymptotic Optimality

CoIT* inherits the theoretical guarantees of sampling-based informed search planners. Because new states are generated from uniform random samples in the informed set, the probability of eventually sampling any region of the free configuration space approaches one as the number of samples tends to infinity. Formally,

$$\lim_{n \to \infty} \mathbb{P}(\mathcal{R}_n \cap X_{\text{goal}} \neq \varnothing) = 1, \tag{10}$$

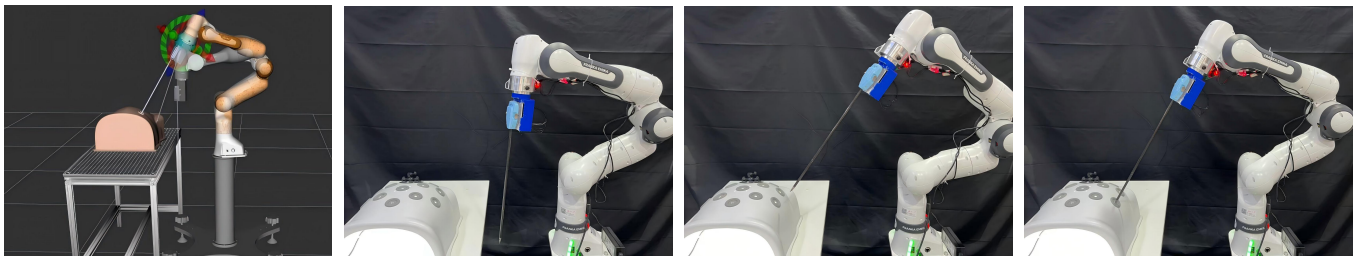
where $\mathcal{R}_n$ denotes the set of states reachable from the start state in the search graph after n samples. Consequently, CoIT* is *probabilistically complete*: if a feasible path exists, the planner will find it with probability approaching one as the number of samples increases.

Asymptotic Optimality: CoIT* preserves the asymptotic optimality guarantee of informed sampling-based planners. Formally, we have

$$\mathbb{P}\left(\limsup_{q \to \infty} \min_{\sigma \in \Sigma_q} \{c(\sigma)\} = c^* \right) = 1. \tag{11}$$

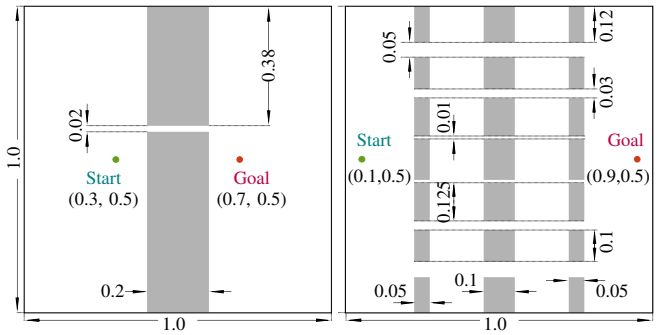
where q is the number of samples, $\Sigma_q \subset \Sigma$ is the set of valid paths found by the planner, $c : \Sigma \to [0, \infty)$ is the cost function, and c^* is the optimal cost.

Notably, when the selected reverse edge comes from Q_{PriR}, after multi-resolution checking, even if the edge is collision-free we do not immediately expand its endpoint. This design choice ensures that Q_R remains strictly ordered. As shown in Alg. 1, line 28, the expansion of Q_R subsequently augments Q_{PriR}, thereby maintaining the admissibility of the heuristic that anchors the overall search. Consequently, the best-cost solution identified by CoIT* converges almost surely to the optimal solution as the number of samples tends to infinity.



(a) Simulation surgical scenery (b) Start configuration (c) Trans. config. - (CoIT*) (d) Goal configuration

Fig. 3: Snapshots of a surgical-robot task with a 7-DoF manipulator: (a) simulation scene with an abdominal phantom and table; (b) start configuration aligned to the trocar entry; (c) transition configuration generated by CoIT during insertion; (d) goal configuration with the instrument inserted and aligned with the target region.



(a) Narrow Passage (NP) (b) Dividing Walls (DW)

Fig. 4: Two 2-D snapshots illustrating the simulated planning benchmarks (Section IV). The configuration space $X \subset \mathbb{R}^n$ is a unit hypercube in all cases. The two benchmarks are the narrow passage (NP) and dividing walls (DW); aggregated results are reported in Fig. 5.

IV. Experimental Results

In this work, we integrate the Cooperative Informed Trees (CoIT*) algorithm into the Open Motion Planning Library (OMPL) [29] and benchmark our planner using the Planner Arena benchmark database [30], the Planner Developer Tools [31], and MoveIt [32]. Evaluations are conducted on a machine equipped with an Intel Core i5-12400 CPU and 32 GB of RAM. CoIT* is tested against existing algorithms in both simulated random environments (Fig. 4) and real-world manipulation problems, shown in Fig. 1 and Fig. 3. The comparison involves official OMPL implementations of RRT-Connect, Informed RRT*, BIT*, AIT*, ABIT*, and EIT*. The primary objective is to minimize the path length c_{init}^{med} within a fixed time budget and to reduce the median initial-solution time t_{init}^{med}. The RGG constant η is set to 1.001 and the rewire factor to 1.1 for all planners. We set $r_{\text{high}} = 31$ as a moderate pre-check level because it aligns with our midpoint-refinement hierarchy, which uses odd checkpoints, and provides reliable screening at modest cost; for example, a much smaller value such as 15 is less discriminative near obstacles, whereas a much larger value such as 63 increases overhead with diminishing returns.

For RRT-based algorithms, a goal bias of 5% was incorporated, with maximum edge lengths of 0.5, 1.25, and 3.0 in $\mathbb{R}^4$, $\mathbb{R}^8$, $\mathbb{R}^{16}$. Batch sampling planners maintained a fixed sampling of 100 states per batch across different dimensionalities of the state space. These planners employed the Euclidean distance and effort as a heuristic, respectively.

A. Simulation Experimental Tasks

The planners were evaluated across three configuration-space dimensions: $\mathbb{R}^4$, $\mathbb{R}^8$, and $\mathbb{R}^{16}$. In the first test environment, we simulated a constrained narrow-passage (NP) with a slim gap of 0.02 that effectively permits valid non-intersecting solutions only in one general direction (Fig. 4a). Each planner was executed 100 times per instance, and the per-benchmark computation times are indicated in the figure labels. The overall success rates and median path lengths for all planners are shown in Fig. 5a, 5c, and 5e. The results indicate that CoIT* finds initial solutions quickly with the shortest times across dimensions, whereas EIT* requires longer to obtain the initial solution.

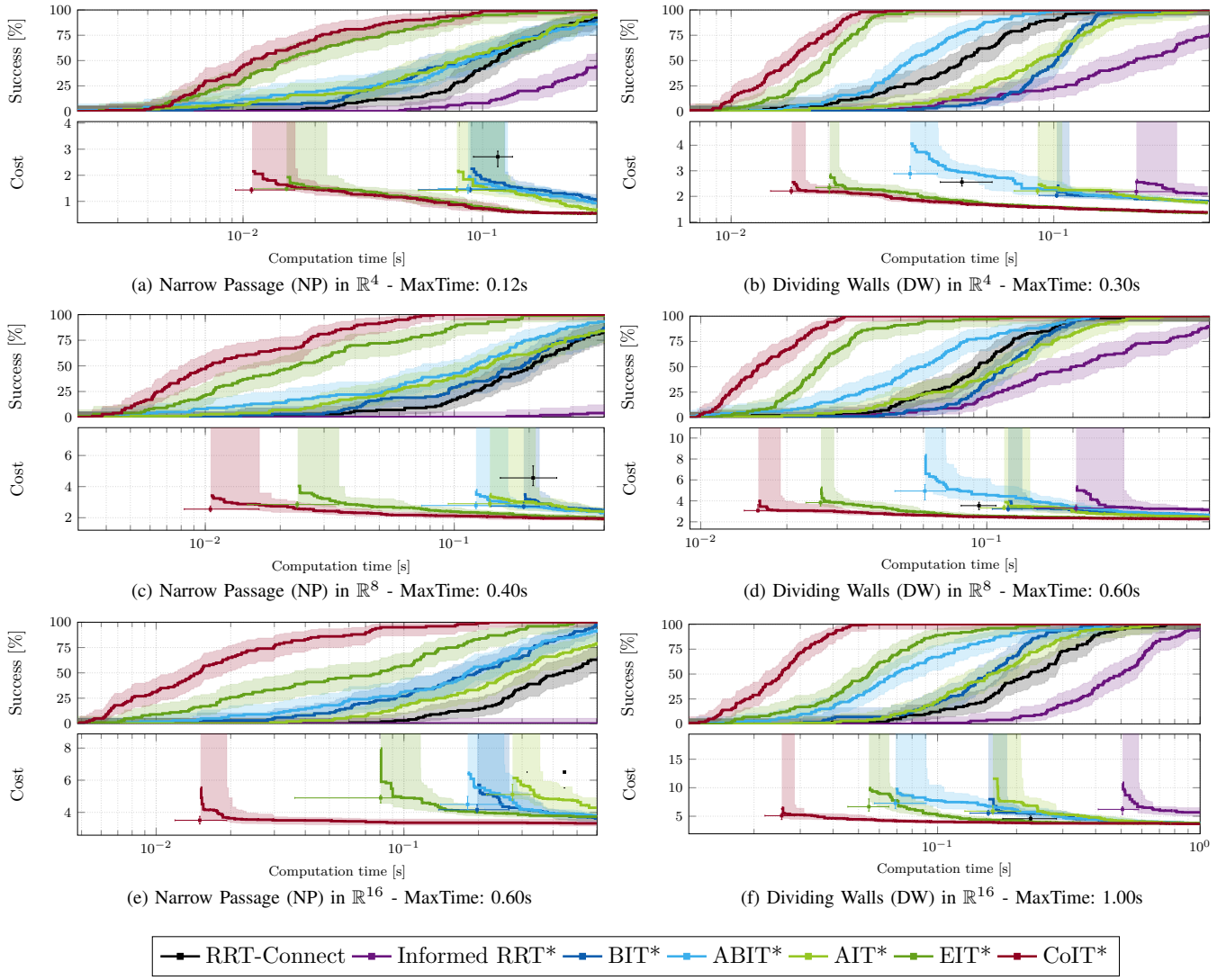
In the second scenario, we simulated a constrained dividing-wall (DW) environment with multiple narrow gaps, which allows all planners to search for optimal paths along several salient directions (Fig. 4b). Each planner was run 100 times with different random seeds, and the maximum computation time of the best-performing planner is indicated in the figure labels. The overall success rates and median path lengths for all planners are shown in Fig. 5b, 5d, and 5f. Experimental results show that CoIT* outperforms EIT* in both computation time and path length across all dimensions.

As shown in Table II, CoIT* consistently reduces the median initial-solution time $t_{\text{init}}^{\text{med}}$ relative to EIT* across all six settings, with reductions ranging from 23.38% to 81.39% (average $\approx 47.00\%$). The gain increases with dimensionality. In NP, it grows from 28.95% in $\mathbb{R}^4$ to 55.13% in $\mathbb{R}^8$ and 81.39% in $\mathbb{R}^{16}$, where $t_{\text{init}}^{\text{med}}$ drops from 0.0806 to 0.0150. In DW, it rises from 23.38% in $\mathbb{R}^4$ to 39.69% in $\mathbb{R}^8$ and 53.47% in $\mathbb{R}^{16}$, with 0.0548 decreasing to 0.0255. Moreover, the cost advantage becomes more pronounced in higher dimensions: in $\mathbb{R}^8$ and $\mathbb{R}^{16}$, CoIT* achieves consistently lower $c_{\text{init}}^{\text{med}}$ and $c_{\text{final}}^{\text{med}}$ than EIT* for both NP and DW.

Overall, CoIT* finds initial solutions substantially faster across dimensions. We attribute these improvements to its *multi-resolution cooperative edge screening*: forward and reverse searches exchange directional information to bias exploration toward the goal, filter less favorable neighbor edges, and defer expensive validations—thereby reducing computation time and accelerating the discovery of the initial solution.

B. Real-world Path Planning Tasks

We compared CoIT* with EIT*, AIT*, and BIT* under identical settings to evaluate their performance on a 7-DoF Franka manipulator over 30 runs, focusing on convergence



(a) Narrow Passage (NP) in $\mathbb{R}^4$ - MaxTime: 0.12s

(b) Dividing Walls (DW) in $\mathbb{R}^4$ - MaxTime: 0.30s

(c) Narrow Passage (NP) in $\mathbb{R}^8$ - MaxTime: 0.40s

(d) Dividing Walls (DW) in $\mathbb{R}^8$ - MaxTime: 0.60s

(e) Narrow Passage (NP) in $\mathbb{R}^{16}$ - MaxTime: 0.60s

(f) Dividing Walls (DW) in $\mathbb{R}^{16}$ - MaxTime: 1.00s

Fig. 5: Detailed experimental results from Section IV-A are summarized above. Panels (a), (c), and (e) depict the narrow-passage (NP) benchmark in $\mathbb{R}^4$, $\mathbb{R}^8$, and $\mathbb{R}^{16}$, respectively; panels (b), (d), and (f) depict the dividing-wall (DW) benchmark in the same dimensions. In the performance plots, box–whisker summaries represent the distributions of solution time and cost, while solid traces show the progression of solution cost for an almost-surely optimal planner (unsuccessful runs are assigned infinite cost). Error bars indicate nonparametric 99% confidence intervals for time and cost.

TABLE II: Benchmarks evaluation comparison (Fig. 5)

	Effort Informed Trees			Cooperative Informed Trees			$t_{init}^{med} \Uparrow$ (%)
	t_{init}^{med}	c_{init}^{med}	c_{final}^{med}	t_{init}^{med}	c_{init}^{med}	c_{final}^{med}	
NP – $\mathbb{R}^4$	0.0152	1.4787	0.5345	0.0108	1.4362	0.5402	28.95
NP – $\mathbb{R}^8$	0.0234	2.8480	1.9938	0.0105	2.5518	1.9225	55.13
NP – $\mathbb{R}^{16}$	0.0806	4.8987	3.7208	0.0150	3.5061	3.3161	**81.39**
DW – $\mathbb{R}^4$	0.0201	2.3475	1.3470	0.0154	2.2099	1.3752	23.38
DW – $\mathbb{R}^8$	0.0262	3.8299	2.2923	0.0158	3.0746	2.2661	39.69
DW – $\mathbb{R}^{16}$	0.0548	6.6715	3.7410	0.0255	5.0945	3.6542	53.47

to optimal solution cost and success rate. The *surgical robot* - ENV in Fig. 1 and Fig. 3 requires avoiding the table and an abdominal phantom, finding an entry through a trocar port, and computing a collision-free path from the start state to the goal region.

Results. With a 3.0 s time budget per run, CoIT* achieved a 97.89% success rate with a median solution cost of 9.34. EIT* attained 91.56% with 10.15, AIT* achieved 86.47% with 13.26, and BIT* achieved 81.88% with 14.02. Throughout the experiments, CoIT* consistently produced higher-quality paths than the other planners.

V. Conclusion

This paper proposes Cooperative Informed Tree (CoIT*), a sampling-based, asymmetric bi-directional planner that integrates a multi-resolution, multi-heuristic queue cooperation mechanism. Through cooperative information exchange between forward and reverse searches and the sharing of global- and local-resolution cues, CoIT* performs adaptive edge screening and lazy validation at both global and local scales, thereby improving the accuracy of the reverse search and reducing the validation time on the forward side. We validate CoIT* on multidimensional benchmark problems as well as simulated and real surgical-robot tasks. The results show that, compared with state-of-the-art planners, CoIT* achieves faster initial path convergence, better path quality, and significantly lower planning time; real-world tasks further demonstrate the method's robustness and practical value.

VI. Acknowledgements

This work was supported by China Mobile Hunan Company Limited and China Mobile Communications Group

Co., Ltd., and was conducted under the project "Networked Robotic System for Major Equipment Manufacturing (5G+Robotics)". This work was supported by the National Natural Science Foundation of China under Grant U22B2050 and 62425305

REFERENCES

[1] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[2] W. Du, F. Islam, and M. Likhachev, "Multi-resolution a," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 11, no. 1, 2020, pp. 29–37.

[3] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, and M. Likhachev, "Multi-heuristic a," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 224–243, 2016.

[4] R. Natarajan, M. Saleem, S. Aine, M. Likhachev, and H. Choset, "A-mha*: anytime multi-heuristic a," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 10, no. 1, 2019, pp. 192–193.

[5] D. M. Saxena, T. Kusnur, and M. Likhachev, "Amra*: Anytime multi-resolution multi-heuristic a," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 3371–3377.

[6] R. Bellman, "Dynamic programming, princeton univ," *Press Princeton, New Jersey*, 1957.

[7] M. P. Strub and J. D. Gammell, "Adaptively informed trees (ait*) and effort informed trees (eit*): Asymmetric bidirectional sampling-based path planning," *The International Journal of Robotics Research*, vol. 41, no. 4, pp. 390–417, 2022.

[8] L. Zhang, K. Cai, Z. Sun, Z. Bing, C. Wang, L. Figueredo, S. Haddadin, and A. Knoll, "Motion planning for robotics: A review for sampling-based planners," *Biomimetic Intelligence and Robotics*, p. 100207, 2025.

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

[10] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," in *Technical Report*. Computer Science Department, Iowa State University, 1998.

[11] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.

[12] 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 *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 2997–3004.

[13] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[14] O. Salzman and D. Halperin, "Asymptotically-optimal motion planning using lower bounds on cost," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4167–4172.

[15] 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," *The International journal of robotics research*, vol. 34, no. 7, pp. 883–921, 2015.

[16] 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 *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 3067–3074.

[17] J. D. Gammell, T. D. Barfoot, and S. S. Srinivasa, "Batch informed trees (BIT*): Informed asymptotically optimal anytime search," *The International Journal of Robotics Research*, vol. 39, no. 5, pp. 543–567, 2020.

[18] M. Penrose, *Random geometric graphs*. OUP Oxford, 2003, vol. 5.

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

[20] M. P. Strub and J. D. Gammell, "Adaptively informed trees (AIT*): Fast asymptotically optimal path planning through adaptive heuristics," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3191–3198.

[21] L. Zhang, K. Chen, K. Cai, Y. Zhang, Y. Dang, Y. Wu, Z. Bing, F. Wu, S. Haddadin, and A. Knoll, "Direction informed trees (dit*): Optimal path planning via direction filter and direction cost heuristic," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 1766–1772.

[22] C. Dellin and S. Srinivasa, "A unifying formalism for shortest path problems with expensive edge evaluations via lazy best-first search over paths with edge selectors," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 26, 2016, pp. 459–467.

[23] A. Mandalika, O. Salzman, and S. Srinivasa, "Lazy receding horizon a* for efficient path planning in graphs with expensive-to-evaluate edges," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 28, 2018, pp. 476–484.

[24] A. Mandalika, S. Choudhury, O. Salzman, and S. Srinivasa, "Generalized lazy search for robot motion planning: Interleaving search and edge evaluation via event-based toggles," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, 2019, pp. 745–753.

[25] D. Yi, R. Thakker, C. Gulino, O. Salzman, and S. Srinivasa, "Generalizing informed sampling for asymptotically-optimal sampling-based kinodynamic planning via markov chain monte carlo," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7063–7070.

[26] Y. Wang, B. Mu, and O. Salzman, "Asymptotically optimal sampling-based motion planning through anytime incremental lazy bidirectional heuristic search," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 1787–1793.

[27] R. Bohlin and L. E. Kavraki, "Path planning using lazy prm," in *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, vol. 1. IEEE, 2000, pp. 521–528.

[28] J. Thayer, J. Benton, and M. Helmert, "Better parameter-free anytime search by minimizing time between solutions," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 3, no. 1, 2012, pp. 120–128.

[29] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[30] M. Moll, I. A. Sucan, and L. E. Kavraki, "Benchmarking motion planning algorithms: An extensible infrastructure for analysis and visualization," *IEEE Robotics & Automation Magazine*, vol. 22, no. 3, pp. 96–102, 2015.

[31] J. D. Gammell, M. P. Strub, and V. N. Hartmann, "Planner developer tools (PDT): Reproducible experiments and statistical analysis for developing and testing motion planners," in *Proceedings of the Workshop on Evaluating Motion Planning Performance (EMPP), IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[32] M. Görner, R. Haschke, H. Ritter, and J. Zhang, "Moveit! task constructor for task-level motion planning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 190–196.