









APT*: Asymptotically optimal motion planning via adaptively prolated elliptical r-nearest neighbors

Authors	Zhang, Liding;Wang, Sicheng;Cai, Kuanqi;Bing, Zhenshan;Wu, Fan;Wang, Chaoqun;Haddadin, Sami;Knoll, Alois
Citation	L. Zhang et al., "APT*: Asymptotically Optimal Motion Planning via Adaptively Prolated Elliptical R-Nearest Neighbors," in IEEE Robotics and Automation Letters, vol. 10, no. 10, pp. 10242-10249, Oct. 2025, doi: 10.1109/LRA.2025.3598616
DOI	10.1109/LRA.2025.3598616
Publisher	IEEE
Download date	2026-05-15 07:20:51
Link to Item	https://hdl.handle.net/20.500.14634/1648

APT*: Asymptotically Optimal Motion Planning via Adaptively Prolated Elliptical R-Nearest Neighbors

Liding Zhang , *Student Member, IEEE*, Sicheng Wang , Kuanqi Cai , *Graduate Student Member, IEEE*,
Zhenshan Bing , *Member, IEEE*, Fan Wu , *Member, IEEE*, Chaoqun Wang , *Member, IEEE*,
Sami Haddadin , *Fellow, IEEE*, and Alois Knoll , *Fellow, IEEE*

I. INTRODUCTION

Abstract—Optimal path planning aims to determine a sequence of states from a start to a goal while accounting for planning objectives. Popular methods often integrate fixed batch sizes and neglect information on obstacles, which is not problem-specific. This study introduces Adaptively Prolated Trees (APT*), a novel sampling-based motion planner that extends based on Force Direction Informed Trees (FDIT*), integrating adaptive batch-sizing and elliptical r -nearest neighbor modules to dynamically modulate the path searching process based on environmental feedback. APT* adjusts batch sizes based on the hypervolume of the informed sets and considers vertices as electric charges that obey Coulomb’s law to define virtual forces via neighbor samples, thereby refining the prolate nearest neighbor selection. These modules employ non-linear prolate methods to adaptively adjust the electric charges of vertices for force definition, thereby improving the convergence rate with lower solution costs. Comparative analyses show that APT* outperforms existing single-query sampling-based planners in dimensions from \mathbb{R}^4 to \mathbb{R}^{16} , and it was further validated through a real-world robot manipulation task.

Index Terms—Sampling-based path planning, elliptical r -nearest neighbor, adaptive batch-size, optimal path planning.

Received 29 May 2025; accepted 31 July 2025. Date of publication 13 August 2025; date of current version 26 August 2025. This article was recommended for publication by Associate Editor J. Ichnowski, and Editor J. Borra’s Sol upon evaluation of the reviewers’ comments. This work was supported in part by the Bavarian State Ministry for Economic Affairs, in part by Regional Development and Energy (StMWi) for the Lighthouse Initiative KI.FABRIK (Phase 1: Infrastructure), in part by Research and Development Project under Grant DIK0249, and in part by the Federal Ministry of Education and Research of Germany (BMBF) through the Programme of “Souverän. Digital. Vernetzt.” Joint Project 6G-life, under Project 16KISK002. (Corresponding author: Zhenshan Bing.)

Liding Zhang, Sicheng Wang, Kuanqi Cai, and Alois Knoll are with the School of Computation, Information and Technology (CIT), Technical University of Munich, 80333 Munich, Germany (e-mail: liding.zhang@tum.de).

Fan Wu is with the School of Computation, Information and Technology (CIT), Technical University of Munich, 80333 Munich, Germany, and also with the School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200072, China.

Sami Haddadin is with the School of Computation, Information and Technology (CIT), Technical University of Munich, 80333 Munich, Germany, and also with the Mohamed bin Zayed University of AI, Abu Dhabi 23201, UAE.

Zhenshan Bing is with the State Key Laboratory for Novel Software Technology and the School of Science and Technology, Nanjing University (Suzhou Campus), Suzhou 215163, China, and also with the School of Computation, Information and Technology (CIT), Technical University of Munich, 80333 Munich, Germany (e-mail: zhenshan.bing@tum.de).

Chaoqun Wang is with the School of Control Science and Engineering, Shandong University, Jinan 250100, China.

A video showcasing our experimental results is available at: <https://youtu.be/gCcUr8LiEw4>

This article has supplementary downloadable material available at <https://doi.org/10.1109/LRA.2025.3598616>, provided by the authors.

Digital Object Identifier 10.1109/LRA.2025.3598616

ROBOT motion planning focuses on computing collision-free paths between start and goal configurations while avoiding obstacles [1]. In high-dimensional spaces, sampling-based algorithms [2] are widely regarded as indispensable for solving the *curse of dimensionality* problem [3]. These algorithms are underpinned by the theory of *random geometric graphs* (RGG) [4], a probabilistic framework modeling randomly distributed networks within the *configuration space* (\mathcal{C} -space). The RGG’s structure is characterized by random sample distribution in n -dimensional \mathcal{C} -space within a predefined radius and proximity-based edge formation, typically using Euclidean metrics. This ensures asymptotic optimality in path planning with increased sample density [5].

A. Related Work

Search-based algorithms like Dijkstra’s [6] compute shortest paths by exhaustively exploring all routes, while A* [7] improves efficiency with heuristic guidance. Sampling-based planners, such as Rapidly-exploring Random Trees (RRT) [8] and Probabilistic Roadmaps (PRM) [9], excel in handling non-convex and high-dimensional spaces. RRT grows a tree incrementally from the start toward the goal, where RRT-Connect [10] enhances efficiency by growing trees from both ends. RRT* [11] improves this by using k -nearest [12] or r -nearest [13] neighbor searches and rewiring, offering anytime solutions with guaranteed optimality. Informed RRT* [14] accelerates convergence rate and path optimization by limiting the search space via elliptical *informed sampling* subsets [15].

Batch Informed Trees (BIT*) [16] integrates Informed RRT* and Fast Marching Trees (FMT*) [12], advancing sampling techniques to group states into a compact *batch*-based implicit RGG. Adaptively Informed Trees (AIT*) [17] and Effort Informed Trees (EIT*) [17] employ asymmetrical *forward-reverse* search with sparse collision checks in reverse search, enhancing exploration efficiency. Flexible Informed Trees (FIT*) [18] updates the batch size of each improved approximation for faster initial convergence. Force Direction Informed Trees (FDIT*) [13] builds upon EIT* and utilizes Coulomb’s law with fixed vertex charge to calculate forces that define elliptical neighbor regions. However, in Coulomb’s law, the magnitude of the force is determined not only by the distance between particles but also by their charges. When applied to path planning, it is crucial to optimize the vertices charge regarding planning phases [19]. Prolating neighbors too early may lead to local optimality, while prolate neighbors too late may result in the loss of problem-specific information.

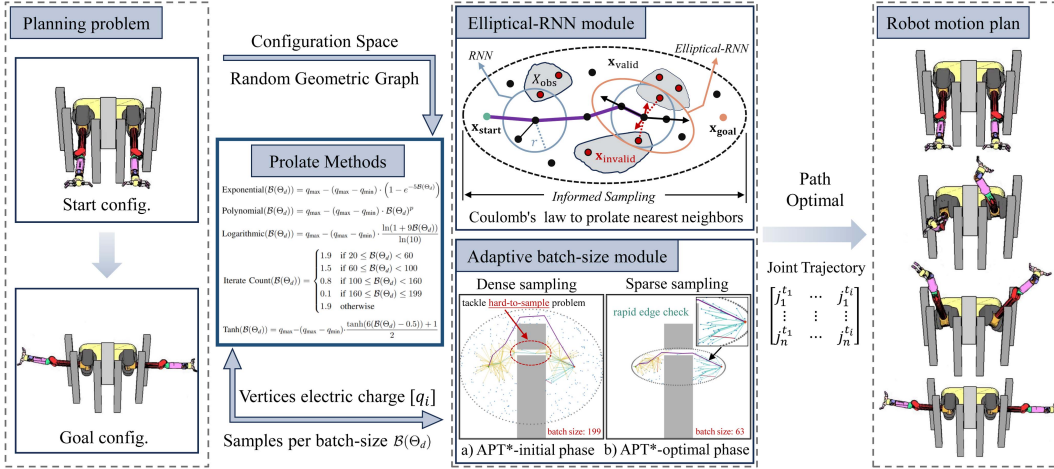


Fig. 1. System diagram of the proposed adaptively prolated method. After defining the start and goal configurations, APT* dynamically adjusts the prolated search region. The framework comprises two primary modules: the Elliptical-RNN module (Fig. 2), which leverages Coulomb's law to compute the forces of valid and invalid nearest neighbor vertices, and the adaptive batch-size module, which optimizes the number of samples per batch based on the Lebesgue measure of the informed sets. These mechanisms ensure that the batch size is adaptively tuned according to the current phase of the search process.

B. Proposed Algorithm and Original Contribution

This letter introduces Adaptively Prolated Trees (APT*), an extension of the sampling-based planner FDIT*, which employs non-linear prolation strategies to dynamically adjust nearest neighbor regions using the adaptive batch-size module. In this module, dense sampling corresponds to the initial phase with a smaller charge, leading to more circular neighbors, while sparse sampling indicates that the path optimization has progressed, allowing a larger charge to increase the Coulomb force's magnitudes, thereby making the neighbors more elliptical. This adjustment impacts the Coulomb force calculation, which is applied to the elliptical r -nearest neighbors (elliptical-RNN) module (Fig. 1), improving search and rewiring efficiency. The approach utilizes the Taylor series expansion of $Tanh$ function to modulate sample charges, directing the search toward valuable regions and enhancing overall pathfinding performance. This work contributes the following extensions to FDIT* and EIT*:

- 1) A non-linear prolate method based on an adaptive batch size module to affect vertices' electric charges, optimizing the Coulomb forces during the planning phase.
- 2) A novel sampling-based path planner APT* that integrates adaptively prolated elliptical-RNN for search and rewiring to quickly obtain high-quality solutions.
- 3) Demonstration of APT*'s effectiveness across various dimensional environments and optimization objectives.

II. PROBLEM FORMULATION AND NOTATION

A. Problem Definition

Optimal Planning: Given a planning problem with state space $X \subseteq \mathbb{R}^n$, where X_{obs} represents the states in collision with obstacles, and $X_{\text{free}} = \text{cl}(X \setminus X_{\text{obs}})$ denotes the set of permissible states, with $\text{cl}(\cdot)$ indicating the *closure* of a set. The initial state is $\mathbf{x}_{\text{start}} \in X_{\text{free}}$, and the goal states are in $X_{\text{goal}} \subset X_{\text{free}}$. A continuous map $\sigma : [0, 1] \mapsto X$ represents a collision-free path, and Σ is the set of all nontrivial paths [11].

The optimal solution, σ^* , is the path that minimizes a chosen cost function $c : \Sigma \mapsto \mathbb{R}_{\geq 0}$. This path connects the initial state

$\mathbf{x}_{\text{start}}$ to a goal state $\mathbf{x}_{\text{goal}} \in X_{\text{goal}}$ through the free space:

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

where $\mathbb{R}_{\geq 0}$ denotes the set of non-negative real numbers. The cost of the optimal path is denoted as c^* . When considering a discrete set of states $X_{\text{samples}} \subset X$, represented as a graph with edges determined by a transition function, its properties can be modeled probabilistically using implicit dense RGGs, where $X_{\text{samples}} = \{\mathbf{x} \sim \mathcal{U}(X)\}$, as discussed in [4].

B. Notation

The state space of the planning problem is denoted by $X \subseteq \mathbb{R}^n$, where $n \in \mathbb{N}$. The start vertex is represented by $\mathbf{x}_{\text{start}} \in X$, and the goals are denoted by $X_{\text{goal}} \subset X$. The coordinates for valid and invalid vertices are given by $\mathbf{x}_{\text{valid}}$ and $\mathbf{x}_{\text{invalid}}$. An admissible estimate for the cost of a path is $\hat{f} : X \rightarrow [0, \infty)$, which characterizes the informed set $X_{\hat{f}}$.

APT*-specific Notation: We assume each vertex has properties of electric charge, denoted as q . These vertices generate either an attractive force, denoted as $\vec{F}_{\text{attractive}}$, or a repulsive force, denoted as $\vec{F}_{\text{repulsive}}$. The Euclidean distance between two vertices is given by r_i , and the unit direction vector between them is denoted by $\hat{\mathbf{r}}_i$. The total virtual Coulomb force acting on the current processing vertex from neighbors, represented as $\vec{F}_{\mathcal{P}}$, is calculated using Coulomb's law and incorporates the elliptical search neighbors, denoted as $V_{\text{ellipseNeighbors}}$. Within this set of neighbors, the ratio of invalid vertices is represented by Φ , while the number of valid and invalid vertices are denoted by N_{valid} and N_{invalid} , respectively. The batch sizes are denoted as $\mathcal{B}(\Theta_d)$, which is tuned by a decay factor Θ_d , with specified minimum m_{min} and maximum m_{max} sample numbers per batch. The Lebesgue measure (i.e., n -dimensional hypervolume) of a prolated hyperellipsoid is represented by $\zeta(c_i, n)$. The non-negative scalar \mathcal{G}_i represents the informed ratio of the Lebesgue measure of hyperellipsoids. τ_t regulates the rate at which the

Algorithm 1: APT* - Elliptical r -nearest neighbors.

Input : \mathbf{x} - The current state, $\mathcal{B}_{\text{adapt}}$ - The adapted batch size, n - The dimensionality

Output: Best set of nearest neighbors $V_{\text{ellipseNeighbors}}$ within an ellipsoidal region around \mathbf{x}

```

1  $V_{\text{ellipseNeighbors}} \leftarrow X_{\text{samples}}$   $\triangleright$  initialize all neighbors
2  $\vec{F}_{\mathcal{D}} \leftarrow \vec{1}, \Phi \leftarrow 1, r \leftarrow \text{getRNNRadius}(\mathcal{B}_{\text{adapt}})$   $\triangleright$  Eq. 2
3 while  $\Phi \geq 0.1$  do  $\triangleright$  neighbor invalid samples less than 10%
4   foreach  $\mathbf{x}_i \in V_{\text{ellipseNeighbors}}$  do  $\triangleright$  update  $\vec{F}_{\mathcal{D}}$ 
5      $r_i \leftarrow \text{distance}(\mathbf{x}, \mathbf{x}_i)$ 
6      $\hat{\mathbf{r}}_i \leftarrow (\mathbf{x}_i - \mathbf{x})/r_i$   $\triangleright$  compute the direction of  $\mathbf{x}_i$ 
7      $q_i \leftarrow \text{getVertexCharge}(\mathcal{B}_{\text{adapt}})$   $\triangleright$  Alg. 3
8     if  $\text{isValid}(\mathbf{x}_i)$  then
9        $\vec{F}_{\text{attractive}} \leftarrow \frac{q_i^2}{r_i^{n-1}} \hat{\mathbf{r}}_i$   $\triangleright$  attractive positive force
10    else
11       $\vec{F}_{\text{repulsive}} \leftarrow -\frac{q_i^2}{r_i^{n-1}} \hat{\mathbf{r}}_i$   $\triangleright$  repulsive negative force
12     $\vec{F}_{\mathcal{D}} \leftarrow \vec{F}_{\mathcal{D}} + \vec{F}_{\text{attractive}} + \vec{F}_{\text{repulsive}}$   $\triangleright$  Eq. 6
13   $N_{\text{total}} \leftarrow 0, N_{\text{invalid}} \leftarrow 0$ 
14  foreach  $\mathbf{x}_i \in V_{\text{ellipseNeighbors}}$  do
15    if  $\text{inEllipse}(\mathbf{x}, \mathbf{x}_i, \vec{F}_{\mathcal{D}}, n)$  then  $\triangleright$  Eq. 9
16       $N_{\text{total}} \leftarrow N_{\text{total}} + 1$ 
17      if not  $\text{isValid}(\mathbf{x}_i)$  then
18         $N_{\text{invalid}} \leftarrow N_{\text{invalid}} + 1$ 
19    else
20       $V_{\text{ellipseNeighbors}} \leftarrow V_{\text{ellipseNeighbors}} \setminus \mathbf{x}_i$ 
21   $\Phi \leftarrow N_{\text{invalid}}/N_{\text{total}}$   $\triangleright$  non-negative charge ratio  $\in (0, 1]$ 
22 foreach  $\mathbf{x}_i \in V_{\text{ellipseNeighbors}}$  do
23   if not  $\text{isValid}(\mathbf{x}_i)$  then
24      $V_{\text{ellipseNeighbors}} \leftarrow V_{\text{ellipseNeighbors}} \setminus \mathbf{x}_i$ 
25 return  $V_{\text{ellipseNeighbors}}$ 

```

ratio decreases. A sigmoid-based smoothing parameter σ_s represents a smoothed value after the decay of the initial and optimal states.

III. ADAPTIVELY PROLATED TREES (APT*)

In this section, we first introduce the concept of the Elliptical-RNN module. Next, we illustrate the adaptive batch-size module. Finally, we propose and evaluate nearest neighbor prolated methods based on non-linear functions.

A. Elliptical R -Nearest Neighbors (RNN) Module

Popular path planners use RNN to traverse the neighbors around the current state, and the radius of nearest neighbors $r(\mathcal{B})$, crucial for defining the graph's sparsity, defined as:

$$r(\mathcal{B}) := 2\eta \left(\left(1 + \frac{1}{n} \right) \left(\frac{\lambda(X_{\hat{f}})}{\lambda(U_{\mathcal{B},n})} \right) \left(\frac{\log(\mathcal{B})}{\mathcal{B}} \right) \right)^{\frac{1}{n}}, \quad (2)$$

where η is a normalization constant, n represents the dimensionality of the space, $\lambda(\cdot)$ denotes the Lebesgue measure, \mathcal{B} is the current batch size, and $U_{\mathcal{B},n}$ is a unit ball in an n -dimensional space [17]. This formula ensures that $r(\mathcal{B})$ adapts to the complexity of the space, enhancing the graph's connectivity without increasing computational effort [20].

We proposed an adaptively prolated method based on the traditional RNN via Coulomb's law. Consider a n -dimensional

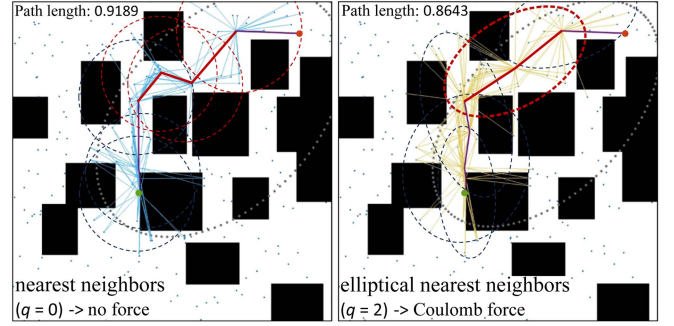


Fig. 2. Illustrations of the 2D representation of elliptical nearest neighbor search. The vertex charge q in Coulomb's law affects the force exerted on elliptical nearest neighbors during the exploration and rewiring phases.

Euclidean space \mathbb{R}^n (Fig. 2). Let two vertexes in \mathcal{C} -space with charge q , separated by a distance r_i . The unit vector from current vertex \mathbf{x} to vertex \mathbf{x}_i is denoted by $\hat{\mathbf{r}}_i$. The n -dimensional Coulomb force is calculated as:

$$\vec{F}_{\mathcal{D}} := \sum_{i=1}^{N_{\text{total}}} \pm k_e \frac{q^2}{r_i^{n-1}} \hat{\mathbf{r}}_i, \quad (3)$$

where $\vec{F}_{\mathcal{D}}$ represents the force vector exerted by the charges on each other, and N_{total} is the number of neighbors around the current state. For a given vertex, we have N_{valid} valid vertices and N_{invalid} invalid vertices in its neighborhood. The vertices in free space exert an attractive force on the neighborhood of the vertex, represented as $\vec{F}_{\text{attractive}}$. Conversely, the surrounding vertices located within obstacles exert a repulsive force on the neighborhood of the vertex, represented as $\vec{F}_{\text{repulsive}}$. These two force vectors can be obtained through the formula below:

$$\vec{F}_{\text{attractive}}(\mathbf{x}) = k_e \sum_{i=1}^{N_{\text{valid}}} \frac{q^2}{\|\mathbf{x} - \mathbf{x}_{\text{valid},i}\|^n} (\mathbf{x}_{\text{valid},i} - \mathbf{x}), \quad (4)$$

$$\vec{F}_{\text{repulsive}}(\mathbf{x}) = -k_e \sum_{j=1}^{N_{\text{invalid}}} \frac{q^2}{\|\mathbf{x} - \mathbf{x}_{\text{invalid},j}\|^n} (\mathbf{x}_{\text{invalid},j} - \mathbf{x}). \quad (5)$$

where k_e is a proportionality constant to Coulomb's constant. q is the charge of the current state, which is dynamically adjusted via the adaptive batch-size module. \mathbf{x} is the position vector of the current state, $\mathbf{x}_{\text{valid},i}$ and $\mathbf{x}_{\text{invalid},j}$ are the position vectors of the i -th valid vertex and the j -th invalid vertex, respectively. n is the dimensionality of the \mathcal{C} -space.

For each vertex \mathbf{x} in the \mathcal{C} -space, all other sampled vertices exert either attractive or repulsive Coulomb-like forces on it. The total force acting on the current vertex is computed as the vector sum of all these interactions:

$$\vec{F}_{\mathcal{D}}(\mathbf{x}) = \vec{F}_{\text{attractive}}(\mathbf{x}) + \vec{F}_{\text{repulsive}}(\mathbf{x}), \quad (6)$$

under the influence of the resultant Coulomb force $\vec{F}_{\mathcal{D}}(\mathbf{x})$, the conventional isotropic RNN ball centered at \mathbf{x} is elongated in the direction of $\vec{F}_{\mathcal{D}}(\mathbf{x})$ and compressed in the orthogonal subspace. The resulting anisotropic neighborhood forms an ellipsoid whose major axis aligns with $\vec{F}_{\mathcal{D}}(\mathbf{x})$; we refer to this prolated search region as the *elliptical-RNN*. To obtain the major-axis direction, the Coulomb force vector is normalized as $\mathbf{u}_1 = \frac{\vec{F}_{\mathcal{D}}(\mathbf{x})}{\|\vec{F}_{\mathcal{D}}(\mathbf{x})\|}$. The Gram-Schmidt orthogonalization (i.e., QR decomposition) [15] is then applied to construct $n - 1$ additional

mutually orthogonal unit vectors $\mathbf{u}_2, \dots, \mathbf{u}_n$, resulting in the orthogonal matrix:

$$\mathbf{Q} = [\mathbf{u}_1 \ \mathbf{u}_2 \ \dots \ \mathbf{u}_n] \in \mathbb{R}^{n \times n}, \quad (7)$$

therefore, the semi-axis is prolonged and aligned with $\vec{F}_{\mathcal{D}}(\mathbf{x})$ to $d_1 = r(1 + k\|\vec{F}_{\mathcal{D}}\|)$, where k is a scaling factor, while keeping the remaining semi-axes at $d_{2..n} = r$. These lengths are inserted into the diagonal scaling matrix:

$$\mathbf{D}^{-2} = \text{diag}(d_1^{-2}, r^{-2}, \dots, r^{-2}), \quad (8)$$

the elliptical-RNN region $X_{\text{eRNN}}(\mathbf{x})$ centered at \mathbf{x} is defined as the set of points $\mathbf{x}_i \in \mathbb{R}^n$ satisfying:

$$X_{\text{eRNN}}(\mathbf{x}) := \{\mathbf{x}_i \in \mathbb{R}^n \mid (\mathbf{x}_i - \mathbf{x})^\top \mathbf{Q} \mathbf{D}^{-2} \mathbf{Q}^\top (\mathbf{x}_i - \mathbf{x}) < 1\}, \quad (9)$$

where the inequality defines the interior of a hyperellipsoid centered at \mathbf{x} , aligned along the direction of the force.

B. Adaptive Batch-Size Module

In this module, we establish the relationship between batch size and the hypervolume of informed sets. Intuitively, larger batch sizes facilitate faster discovery of an initial feasible path, while smaller batch sizes enhance optimization by minimizing edge checks and computational costs (Alg. 2). The batch size \mathcal{B} is dynamically adjusted each time by the following formula:

$$\mathcal{B}(\Theta_d) := \lfloor m_{\min} + \Theta_d \times (m_{\max} - m_{\min}) \rfloor, \quad (10)$$

where $\lfloor \cdot \rfloor$ is the floor function, the decay factor $\Theta_d \in (0, 1)$ is logarithmically smoothed to prevent abrupt transitions. When Θ_d approaches 1, \mathcal{B} nears the m_{\max} , facilitating a faster initial solution. Conversely, when Θ_d approaches 0, \mathcal{B} is closer to the m_{\min} , promoting more efficient path optimization:

$$\Theta_d = \frac{\ln(\tau_t \times \sigma_s + 1)}{\ln(\tau_t + 1)}. \quad (11)$$

where the tuning parameter $\tau_t = (m_{\max} + m_{\min})/n_{\text{dim}}$ is related to the problem domain's dimensionality. The smoothing parameter σ_s via the sigmoid function (without overflow form) is defined as:

$$\sigma_s = \begin{cases} e^{10(\mathcal{G}_i - 0.5)} / (1 + e^{10(\mathcal{G}_i - 0.5)}), & \text{if } \mathcal{G}_i < 0.5 \\ 1 / (1 + e^{-10(\mathcal{G}_i - 0.5)}), & \text{otherwise} \end{cases}, \quad (12)$$

where e is Euler's number $e = \sum_{i=0}^{\infty} \frac{1}{n!}$. This ensures gradual transitions in batch sizes, correlating with informed ratio changes. The informed ratio \mathcal{G}_i , described as:

$$\mathcal{G}_i = \frac{\zeta(c_{\text{current}}, n)}{\zeta(c_{\text{initial}}, n)}, \quad (13)$$

where $\zeta(\cdot, \cdot)$ is the hypervolume contraction as solution cost improves to adjust the batch-size. This ratio \mathcal{G}_i bounded within $(0, 1]$, it ensures batch sizes adapt to the problem's optimization phase. The Lebesgue measure (i.e., hypervolume) of n -dimensional hyper ellipsoid $\zeta(c_i, n)$ is defined as:

$$\zeta(c_i, n) = \frac{\pi^{\frac{n}{2}} c_i (c_i^2 - c_{\min}^2)^{\frac{n-1}{2}}}{2^n \cdot \Gamma(\frac{n}{2} + 1)}. \quad (14)$$

where $\Gamma(\cdot)$ is the gamma function, an extension of factorials to real numbers, c_{\min} denotes the Euclidean distance between $\mathbf{x}_{\text{start}}$ and \mathbf{x}_{goal} , c_i is the current cost of solution.

C. Non-Linear Prolate Methods

In this section, we evaluated five non-linear prolation methods based on the results from the force-based elliptical-RNN module

TABLE I
NON-LINEAR PROLATION METHODS COMPARISON (100 RUNS)

	initial time			initial cost			success
	t_{init}^{\min}	$t_{\text{init}}^{\text{med}}$	t_{init}^{\max}	c_{init}^{\min}	$c_{\text{init}}^{\text{med}}$	c_{init}^{\max}	
EIT* [17]	0.0490	0.0768	∞	1.6361	2.4268	∞	0.92
FIT* [18]	0.0484	0.0688	∞	1.5945	2.1871	∞	0.95
FDIT* [13]	0.0494	0.0658	∞	1.5498	1.9772	∞	0.97
APT*-E	0.0473	0.0650	0.1032	1.5442	1.9405	2.7100	1.00
APT*-P	0.0437	0.0647	∞	1.5148	1.9259	∞	0.97
APT*-L	0.0482	0.0654	0.1110	1.5334	1.8941	2.7410	1.00
APT*-I	0.0472	0.0657	∞	1.6337	1.9161	∞	0.98
APT*-T							
$\alpha = 10$	0.0439	0.0633	0.1075	1.5212	1.9120	2.2960	1.00
$\alpha = 100$	0.0441	0.0639	0.1056	1.4772	1.8826	2.2104	1.00
$\alpha = 1000$	0.0454	0.0648	∞	1.4657	1.8719	∞	0.99

Algorithm 2: APT* - Adaptive batch-size.

Input : last cost c_{last} , current cost c_{current} , dimensionality n
Output: $\mathcal{B}_{\text{adapt}}$ - The adapted batch-size in informed set

- 1 $m_{\min} \leftarrow 1, m_{\max} \leftarrow 2m_{\text{current}} - m_{\min}, \mathcal{B}_{\text{adapt}} \leftarrow 0$
- 2 $\tau_t \leftarrow \text{calTuningParam}(m_{\max}, m_{\min})$
- 3 **if** $c_{\text{current}} \neq c_{\text{last}}$ **or** $c_{\text{last}} := \infty$
- 4 $c_{\text{last}} \leftarrow \text{isCostBetter}(c_{\text{current}})$
- 5 **if pragma once**
- 6 $\zeta_{\text{initial}} \leftarrow \text{lebMeasure}(c_{\text{last}}, n)$ \triangleright initial measure
- 7 $\zeta_{\text{current}} \leftarrow \text{lebMeasure}(c_{\text{current}}, n)$ \triangleright updated measure
- 8 $\mathcal{G}_i \leftarrow \text{informedRatio}(\zeta_{\text{current}}, \zeta_{\text{initial}})$ \triangleright Eq. 13
- 9 $\sigma_s \leftarrow \text{sigmoidSmooth}(\mathcal{G}_i)$ \triangleright Eq. 12
- 10 $\Theta_d \leftarrow \text{calDecayFac}(\sigma_s, \tau_t)$ \triangleright Eq. 11
- 11 $\mathcal{B}_{\text{adapt}} \leftarrow \lfloor m_{\min} + \Theta_d \cdot (m_{\max} - m_{\min}) \rfloor$
- 12 **return** $\mathcal{B}_{\text{adapt}}$

Algorithm 3: APT* - Calculate vertex charge.

Input: $\mathcal{B}_{\text{adapt}}$ - The current adapted batch-size
Output: q_i - The suitable charge of the vertex

- 1: $q_{\min} \leftarrow 0.1, q_{\max} \leftarrow 1.9, q_i \leftarrow q_{\min}$
- 2: $\epsilon \leftarrow 6, \beta \leftarrow -0.5$ $\triangleright \epsilon$ and β are normalization constant
- 3: $\mathcal{B}_{\text{biased}} \leftarrow \epsilon \cdot (\text{normalize}(\mathcal{B}_{\text{adapt}}) + \beta)$
- 4: $q_i \leftarrow \text{tanhProlateMethod}(\mathcal{B}_{\text{biased}}, q_{\min}, q_{\max})$ \triangleright Eq. 15
- 5: **return** q_i

(Sec. III-A) and dynamic batch-size tuning module (Sec. III-B). APT* adaptively adjusts the electric charge of each vertex based on the batch size, thereby influencing the magnitude of the force calculated using Coulomb's law. When the vertex charge increases, the resulting force intensifies, leading to a more pronounced hyperelliptical shape (i.e., higher eccentricity) of the neighbor region. Conversely, when the charge is set to zero, the force becomes null, and the neighbor region assumes a hypersphere shape (i.e., zero eccentricity).

As shown in Table I, t_{init}^{\min} denotes the minimal initial planning time, and c_{init}^{\max} represents the maximal initial cost of the planning problem. For failed attempts, the *cost* and *time* metrics are denoted as infinity. Each method was evaluated across 100 independent runs to analyze its effectiveness in shaping the batch-size \mathcal{B} . Our comparative study investigated various non-linear prolate strategies to understand their distinctive impacts (Fig. 3). In Alg. 3, the scaling factors ϵ and β facilitate the necessary transformation to align the corresponding function regarding the range of vertex charge q and batch size \mathcal{B} . The evaluation considered five approaches: *Exponential* (APT*-E),

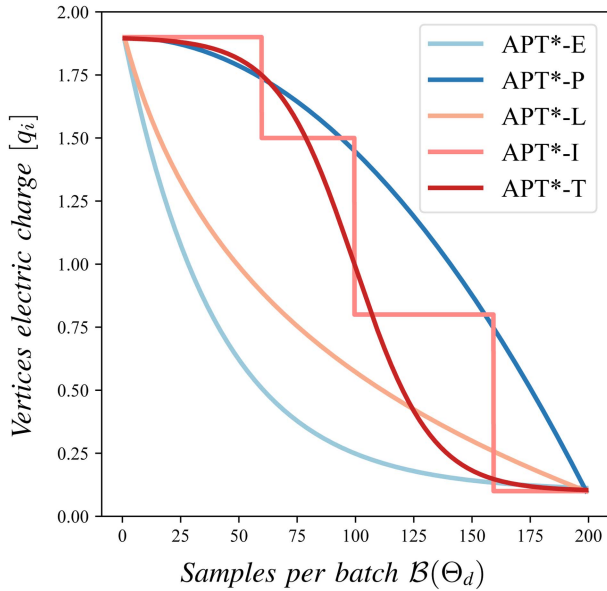


Fig. 3. This graph illustrates the comparison of the five non-linear prolate methods. The maximal electric charge is 1.9, while the minimal charge is 0.1. Similarly, the maximal batch size is 199, and the minimal batch size is 1.

Polynomial (APT*-P), *Logarithmic* (APT*-L), and *Iteration-based* (APT*-I), *Tanh* function (APT*-T).

Experimental results from Table I indicate that APT*-T consistently outperforms the other methods, achieving the lowest *median time* and *median cost* for initial solutions. The computational efficiency of the vertex charge in the APT*-T method is defined by the Taylor expansion of the *Tanh* function, expressed as:

$$q(\mathcal{B}) := \frac{(q_{min} + q_{max})}{2} + \sum_{i=1}^{\alpha} \frac{2^{2i-1} B_{2i} (2^{2i} - 1) \mathcal{B}(\Theta_d)^{2i-1} (q_{min} - q_{max})}{(2i)!}, \quad (15)$$

where $B_{2i} := \sum_{j=0}^i \sum_{k=0}^j (-1)^k \binom{j}{k} \frac{k^{2i}}{j+1}$ denotes as the $2i^{th}$ Bernoulli number and α determine the order of the Taylor series expansion. As shown in Table I, a larger α (e.g., 1000) increases computation time but yields more accurate solutions. Conversely, a smaller α (e.g., 10) reduces pathfinding time at the cost of lower path quality. The electric charge of vertices, $q(\mathcal{B})$, greatly optimizes the total magnitude of Coulomb force $\vec{F}_{\mathcal{D}}$, acting on neighbors. This force plays a crucial role in adjusting the eccentricity of elliptical-RNN. The n -th order eccentricity of the hyperellipsoid is defined as:

$$e_n := 0 = \lim_{\vec{F}_{\mathcal{D}}(q) \rightarrow 0} \sqrt{1 - \frac{r(\mathcal{B})}{(\prod_{i=1}^n d_i)^{1/n}}}, \quad (16)$$

where the eccentricity e_n measure uses the normalized geometric mean. When the electric charge q approaches q_{min} (i.e., $\vec{F}_{\mathcal{D}} \rightarrow 0$), the length of the stretching axis d_1 approaches neighbor radius $r(\mathcal{B})$, thereby e_n of the elliptical-RNN approaches zero, corresponding to a hypersphere.

IV. EXPERIMENTS

In this letter, we utilize the Planner-Arena benchmark database [21], the Planner Developer Tools (PDT) [22], and MoveIt [23] to benchmark motion planner behaviors. APT* was evaluated against popular algorithms in both simulated random scenarios (Fig. 5), path planning problems for dual-Barrett Whole-Arm Manipulator (dual-WAM- \mathbb{R}^{14}) in the Open Robotics Automation Virtual Environment (OpenRAVE, Fig. 4) [24] and real-world manipulation problems (Fig. 7). The comparison involved RRT-Connect, Informed RRT*, BIT*, AIT*, EIT*, FIT*, and FDIT* sourced from the Open Motion Planning Library (OMPL) [25]. These comparisons were tested in simulated tasks of dimensions \mathbb{R}^4 to \mathbb{R}^{16} . The primary objective for all the planners was to minimize path length (cost). 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} . The RGG constant η was uniformly set to 1.001, and the rewiring factor was set to 1.2 for all planners. The implementation of APT* planner into OMPL framework is available at: https://github.com/Liding-Zhang/ompl_apt.git

A. Simulation Experimental Tasks

The planners were evaluated across three distinct benchmarks in three domains: \mathbb{R}^4 , \mathbb{R}^8 , and \mathbb{R}^{16} . In the first scenario, a constrained environment resembling a dividing wall (DW) with several narrow gaps was simulated, allowing valid paths in multiple general directions for non-intersecting solutions (Fig. 5(a)). Each planner was tested over 100 runs, with computation times for each anytime asymptotically optimal planner detailed in the labels, using varying random seeds. The overall success rates and median path lengths for all planners are presented in Fig. 6(a), 6(c), and 6(e). Results indicate that APT* quickly finds an initial solution in all dimensions with minimal time, whereas other planners require more time.

In the second test scenario, random widths were assigned to *axis-aligned hyperrectangles*, generated arbitrarily within the \mathcal{C} -space (Fig. 5(b)). Random rectangle (RR) problems were created for each dimension of the \mathcal{C} -space, with each planner undergoing 100 runs for every instance. Fig. 6(b), Fig. 6(d), and Fig. 6(f) illustrate the proposed method has the highest success rates and lowest median path costs within the computation time compared with other planners. In the final test scenario, we evaluated APT* in the dual-WAM *cage*-ENV for handling intrinsic settings (Fig. 4). The *cage* problems were conducted in \mathbb{R}^{14} , with each planner tested over 100 runs in 5 seconds for every instance. Table II illustrates that the adaptively prolated approach achieved the highest initial convergence rate and fastest computation time.

As observed in Table III, there is a noticeable improvement in median initial times across varied benchmark scenarios, correlating with dimensionality. For instance, in the DW - \mathbb{R}^4 scenario, APT* demonstrates a faster initial median time (i.e., the median value over 100 trials) of maximal 28.96% compared to FDIT* and FIT*. This trend persists across other scenarios, such as RR - \mathbb{R}^8 and Cage - \mathbb{R}^{14} , where APT* consistently convergence faster, including an approximately 30.52% improvement in the *cage* scenario compared to FDIT*.

Overall, APT* achieves superior performance compared to other motion planners. This improvement is attributed to its use of adaptively prolated elliptical-RNN search regions.

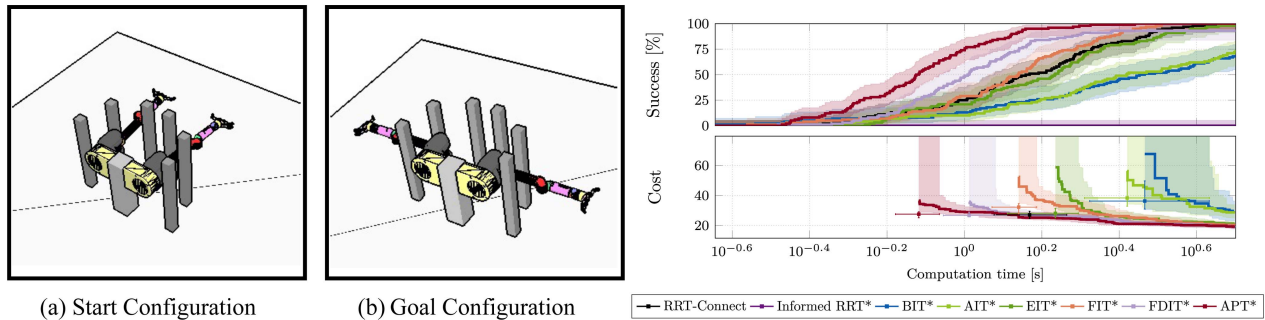


Fig. 4. Illustrations of the dual-arm manipulator (\mathbb{R}^{14}) problem in the *cage-ENV*. Fig. (a) depicts the start configuration of the arms in an extended forward position within a constrained space. Fig. (b) presents the goal configuration, where the arms extend outward in opposite directions without colliding with the cage. Detailed experimental results are presented above and in Table II. All planners have a maximum time of 5 seconds over 100 runs to solve this problem.

TABLE II
PERFORMANCE COMPARISON IN *CAGE-ENV* (FIG. 4) OVER 100 RUNS

Planner	t_{init}^{min}	t_{init}^{med}	t_{init}^{max}	c_{init}^{min}	c_{init}^{med}	c_{init}^{max}	c_{final}^{min}	c_{final}^{med}	c_{final}^{max}	Success
RRT-Connect [10]	0.4264	1.4739	∞	13.1566	27.0866	∞	13.1566	27.0866	∞	0.98
Informed RRT* [14]	∞	∞	∞	∞	∞	∞	∞	∞	∞	0.00
BIT* [16]	0.2270	2.9238	∞	13.6402	36.2940	∞	10.9949	28.6882	∞	0.68
AIT* [17]	0.3836	2.6334	∞	16.4872	38.3468	∞	11.5357	28.6952	∞	0.73
EIT* [17]	0.5533	1.7184	∞	13.3572	28.1140	∞	11.4210	20.8505	∞	0.98
FIT* [18]	0.2749	1.3814	3.2350	12.0491	32.2499	68.6607	11.9310	20.3170	44.4034	1.00
FDIT* [13]	0.3298	1.0828	∞	12.1402	26.7375	∞	10.7653	18.9521	∞	0.93
APT* (ours)	0.3377	0.7523	2.7564	13.3200	27.5476	43.4205	11.0353	19.1839	39.3813	1.00

Here, t and c denote time and cost; i_{init} and f_{final} refer to initial and final solutions; m_{in} , m_{ed} , and m_{ax} represent minimum, median, and maximum values. **Bold** indicates the best in each column. (Unsuccessful runs incur infinite time and cost.)

TABLE III
BENCHMARK EVALUATIONS (\uparrow AND COLOR REPRESENT THE CORRESPONDING MEDIUM INITIAL TIME IMPROVEMENT) (FIG. 6 AND FIG. 4)

	Flexible Informed Trees			Force Direction Informed Trees			Adaptively Prolated Trees (ours)			$t_{init}^{med} \uparrow / \uparrow$ (%)
	t_{init}^{med}	c_{init}^{med}	c_{final}^{med}	t_{init}^{med}	c_{init}^{med}	c_{final}^{med}	t_{init}^{med}	c_{init}^{med}	c_{final}^{med}	
DW — \mathbb{R}^4	0.0252	2.2646	1.5143	0.0308	2.1067	1.5268	0.0179	1.7210	1.2394	28.96 / 21.49
DW — \mathbb{R}^8	0.0334	3.5209	2.3365	0.0288	3.2466	2.3093	0.0244	2.6658	1.7006	26.94 / 20.77
DW — \mathbb{R}^{16}	0.0568	6.3452	3.7953	0.0563	5.9502	3.6954	0.0374	3.7873	2.7854	34.15 / 33.57
RR — \mathbb{R}^4	0.0603	1.7762	1.5929	0.0594	1.7050	1.5616	0.0538	1.7725	1.5597	10.77 / 9.43
RR — \mathbb{R}^8	0.1166	3.4637	2.5049	0.1078	3.3235	2.4449	0.0949	2.8425	2.1976	18.61 / 11.96
RR — \mathbb{R}^{16}	0.1780	5.1412	3.8146	0.1513	5.0367	3.8190	0.1183	3.8367	3.2146	33.54 / 21.81
Cage — \mathbb{R}^{14}	1.3814	32.2499	20.3170	1.0828	26.7375	19.1839	0.7523	27.5476	18.9521	45.54 / 30.52

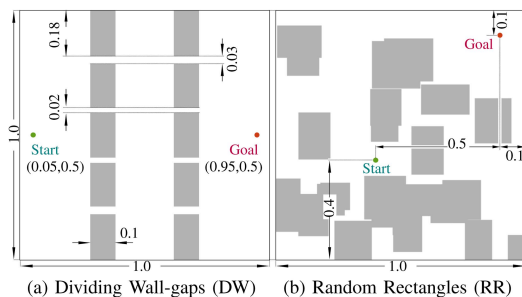


Fig. 5. Simulated planning problems were visualized using a 2D representation. The possible states, represented as $X \subset \mathbb{R}^n$, were confined within a hypercube of side length 1.0 in both scenarios. Ten variations of the dividing wall gaps and random rectangles experiment were conducted, with the results presented in Fig. 6.

B. Real-World Path Planning Tasks

To evaluate the algorithm's performance in real-world scenarios, two numerical experiments are conducted on a base-manipulator (DARKO) platform to demonstrate the efficiency and extensibility of the proposed path planner. We evaluate APT* alongside optimal path planners FIT* and FDIT*, comparing their performance in converging to the optimal solution cost and success rate over 30 runs. A collision-free path connecting the start state to the goal region is required. APT* demonstrated its effective prolation method during The *kitchen printer-ENV* features manipulate 3D-printed kitchen tools, while the *Shelf-ENV* environment involves navigating through industry standard narrow spaces (Fig. 7).

1) *Kitchen Model Tool Printing Task*: In the first task, we utilized the DARKO robot positioned in front of the kitchen and the 3D printer model. The start and goal configurations are

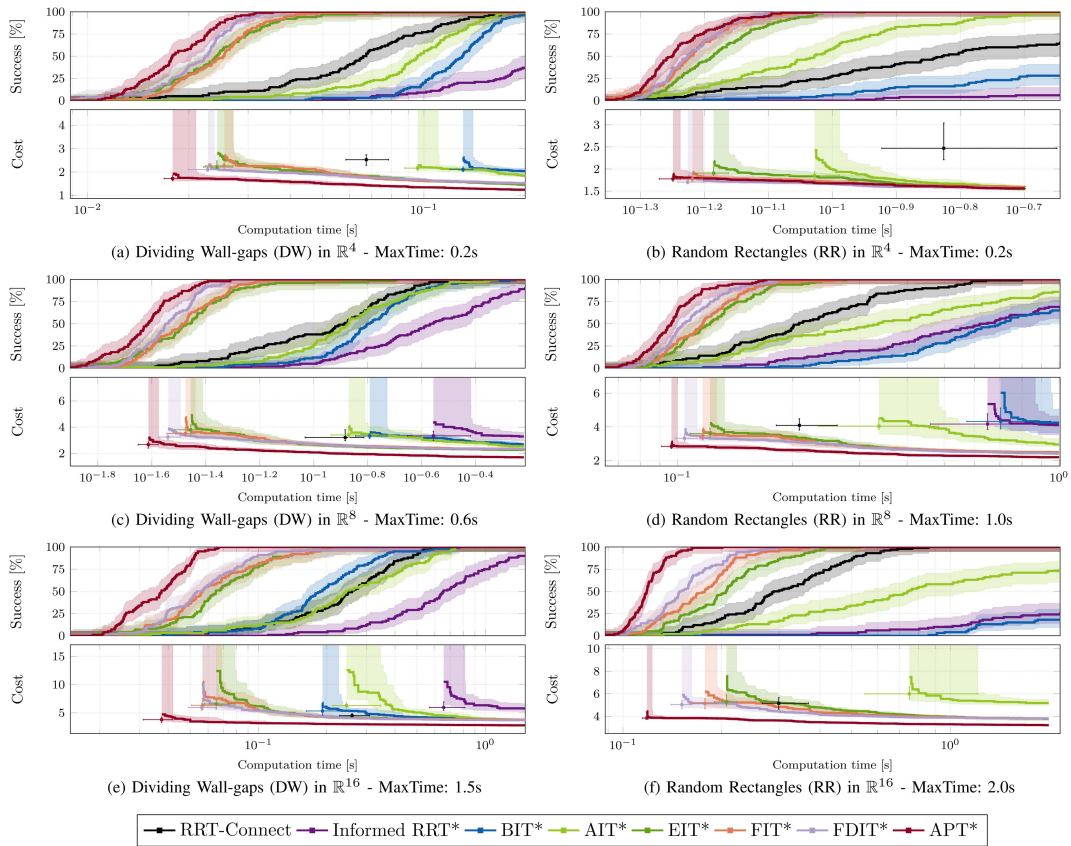


Fig. 6. Detailed experimental results from Section IV-A are presented above. MaxTime is the planner’s maximum allotted planning time. Fig. (a), (c), and (e) depict test benchmark dividing walls (DW) outcomes in \mathbb{R}^4 to \mathbb{R}^{16} , respectively. Panel (b) showcases random rectangle (RR) experiments in \mathbb{R}^4 , while panels (d) and (f) demonstrate in \mathbb{R}^8 and \mathbb{R}^{16} . In the cost plots, boxes represent solution cost and time, with lines showing cost progression for optimal planners (unsuccessful runs have infinite costs). Error bars provide nonparametric 99% confidence intervals for solution cost and time.

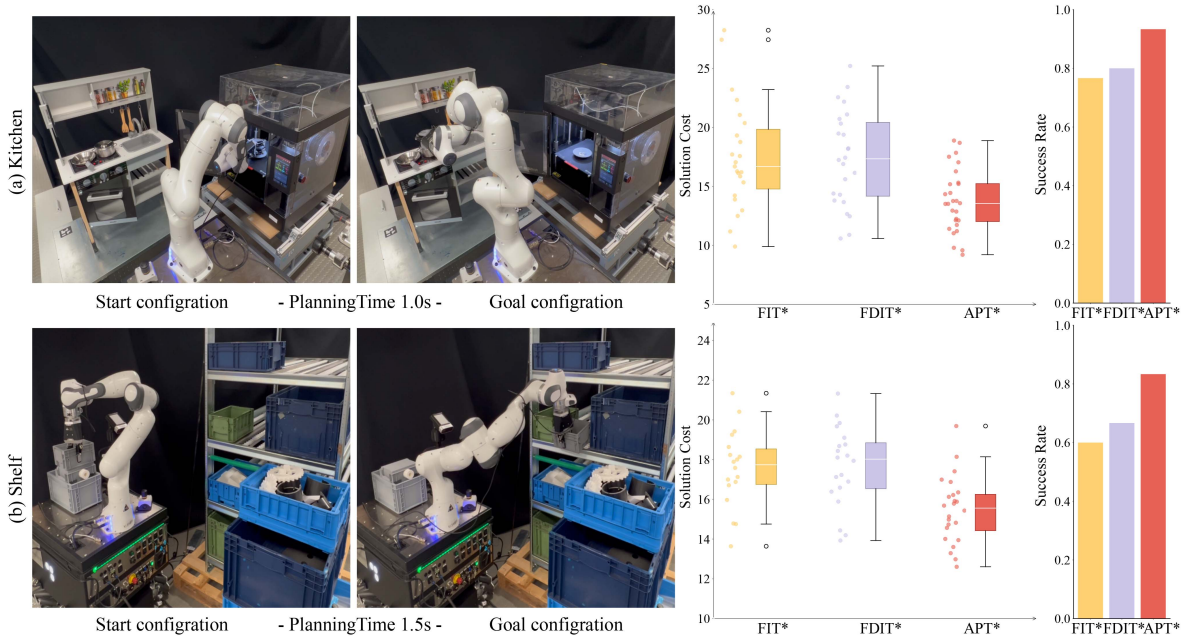


Fig. 7. Experimental results from Section IV-B are summarized above. Fig. 7(a) shows the *kitchen*-ENV with a manipulator taking printed parts from the printer to the kitchen model. Fig. 7(b) highlights the *shelf*-ENV, showing the start/goal configurations for handling an industry-standard (tolerance ± 5 mm) container. Cost box plots display solution costs per planner, with white lines indicating mean cost progression (unsuccessful runs assigned infinite cost).

illustrated in Fig. 7(a). This task is particularly challenging as the manipulator must navigate the geometric shape of the printed tool within a cluttered space while also avoiding collisions between the base robot, printer wall, and the kitchen shelves. The complexity is further heightened by the need for precise movements in a confined space. Each planner was allotted 1.0 seconds to solve this kitchen tool print reallocation problem. Over the course of 30 trials, FIT* managed a 76.66% success rate with a median solution cost of 16.2743. FDIT* had a success rate of 80% with a median solution cost of 16.7816. APT* has the highest success rate of 93.33% and the lowest average solution cost of 13.6948.

2) *Industry Shelf Container Rearrangement Task*: The initial and final configurations for the shelf task are depicted in Fig. 7(b). This task involves carrying an industry-standard container from the base robot and repositioning it on the third shelf layer between two containers. Due to component standardization (tolerance ≤ 5 mm), the challenge lies in precisely inserting industry containers into narrow spaces. The task aims to place the industry-standard container between two larger containers on the shelf, making the planning of a collision-free feasible path particularly difficult. Each planner was allocated 1.5 seconds to solve this confined, limited space carry-on and insertion problem. Across 30 trials, FIT* managed a 60% success rate with a median solution cost of 17.7357. FDIT* had a 66.67% success rate with a median solution cost of 18.0254. APT* achieved the highest success rate of 83.33% with the best median solution cost of 15.7516.

In short, compared with the FIT* and the FDIT*, the APT* achieves the best performance in finding the initial solution and converging to the optimal solution.

V. CONCLUSION AND FUTURE WORK

In this letter, we introduce the Adaptively Prolated Trees (APT*), an extension of the Force Direction Informed Trees (FDIT*) [13] that employs a non-linear prolation-based nearest neighbor approach. APT* assigns electric charge properties to vertices and utilizes an adaptive batch-size module to optimize them. These charges determine the Coulomb force magnitude via Coulomb's law, adjusting the prolated distance for elliptical r -nearest neighbors. APT* optimizes the prolated distances to prevent premature convergence to local minima and adjusts the nearest explore region during path optimization. Future research could focus on integrating human awareness and safety constraints into APT* for planning in dynamic scenarios (i.e., via local motion planners) and utilizing single instruction/multiple data parallelism methods [26] to reduce computation effort, enhancing overall planning efficiency.

REFERENCES

[1] J. D. Gammell and M. P. Strub, "Asymptotically optimal sampling-based motion planning methods," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 4, pp. 295–318, 2021.

[2] A. Orthey, C. Chamzas, and L. E. Kavraki, "Sampling-based motion planning: A comparative review," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 7, no. 1, pp. 285–310, 2024.

[3] L. Zhang et al., "Motion planning for robotics: A review for sampling-based planners," *Biomimetic Intell. Robot.*, vol. 5, no. 1, 2025, Art. no. 100207.

[4] M. Penrose, *Random Geometric Graphs*, vol. 5. Oxford, U.K.: Oxford Univ. Press, 2003.

[5] L. Zhang et al., "Estimated informed anytime search for sampling-based planning via adaptive sampler," *IEEE Trans. Automat. Sci. Eng.*, vol. 22, pp. 18580–18593, 2025.

[6] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[7] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.

[8] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Res. Rep. 9811, 1998.

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

[10] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. ICRA. Millennium Conf. IEEE Int. Conf. Robot. Automat. Symposia Proc.*, 2000, vol. 2, pp. 995–1001.

[11] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011.

[12] 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," *Int. J. Robot. Res.*, vol. 34, no. 7, pp. 883–921, 2015.

[13] L. Zhang et al., "Elliptical K-nearest neighbors: Path optimization via Coulomb's law and invalid vertices in C-space obstacles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 12032–12039, 2024.

[14] 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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 2997–3004.

[15] J. D. Gammell, T. D. Barfoot, and S.S. Srinivasa, "Informed sampling for asymptotically optimal path planning," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 966–984, Aug. 2018.

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

[17] M. P. Strub and J. D. Gammell, "Adaptively informed trees (AIT*) and effort informed trees (EIT*): Asymmetric bidirectional sampling-based path planning," *Int. J. Robot. Res.*, vol. 41, no. 4, pp. 390–417, 2022.

[18] L. Zhang et al., "Flexible informed trees (FIT*): Adaptive batch-size approach in informed sampling-based path planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2024, pp. 3146–3152.

[19] L. Zhang, Y. Ling, Z. Bing, F. Wu, S. Haddadin, and A. Knoll, "Tree-based grafting approach for bidirectional motion planning with local subsets optimization," *IEEE Robot. Automat. Lett.*, vol. 10, no. 6, pp. 5815–5822, Jun. 2025.

[20] K. Solovey and M. Kleinbort, "The critical radius in sampling-based motion planning," *Int. J. Robot. Res.*, vol. 39, no. 2–3, pp. 266–285, 2020.

[21] M. Moll, I. A. Sucas, and L. E. Kavraki, "Benchmarking motion planning algorithms: An extensible infrastructure for analysis and visualization," *IEEE Robot. Automat. Mag.*, vol. 22, no. 3, pp. 96–102, Sep. 2015.

[22] 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 *Proc. Workshop Evaluating Motion Plan. Perform., IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 1–4.

[23] M. Görner, R. Haschke, H. Ritter, and J. Zhang, "Moveit! task constructor for task-level motion planning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 190–196.

[24] R. Diankov and J. J. Kuffner, "OpenRAVE: A planning architecture for autonomous robotics," Carnegie Mellon Univ., Tech. Rep. CMU-RI-TR-08-34 79, 2008.

[25] I. A. Sucas, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robot. Automat. Mag.*, vol. 19, no. 4, pp. 72–82, Dec. 2012.

[26] T. S. Wilson, W. Thomason, Z. Kingston, L. E. Kavraki, and J. D. Gammell, "Nearest-neighbourless asymptotically optimal motion planning with Fully Connected Informed Trees (FCIT*)," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2025.