WiTAH A*: Winding-Constrained Anytime Heuristic Search for a Pair of Tethered Robots

Xingjian Xue, Sze Zheng Yong

***Abstract*— In this paper, we propose a variant of the anytime hybrid A* algorithm that generates a fast but suboptimal solution before progressively optimizing the paths to find the shortest winding-constrained paths for a pair of tethered robots under curvature constraints. Specifically, our proposed algorithm uses a tangent graph as its underlying search graph and leverages an anytime A* search framework with appropriately defined cost metrics in order to reduce the overall computation and to ensure that a winding angle constraint is satisfied. Moreover, we prove the completeness and optimality of the algorithm for finding the shortest winding-constrained paths in an anytime fashion. The effectiveness of the proposed algorithm is demonstrated via simulation experiments.**

I. Introduction

Tethered robots, such as TReX [1], [2] or DuAxel [3], are designed to navigate challenging terrains, including steep slopes during planetary missions, utilizing the tension generated by their tethers. For these robots to effectively explore such extreme environments, particularly steep slopes, it is crucial to provide sufficient tension in the tether to support their weight. However, in certain scenarios, it is not always possible to achieve the needed tension due to hardware limitations. In these cases, the tether can be wound around environmental objects to leverage the capstan effect, where the resulting friction enhances the tension provided by the tether, thereby aiding in the robots' navigation on steep inclines.

Motion planning for a single tethered robot [4]–[9] and multiple tethered robots [10], [11] has been extensively studied in recent years. In the works of [4], [5], the authors presented an algorithm that identified a path within the same homotopy class for a tethered robot to move on steep terrains without the risk of tether entanglement, utilizing a boundary triangulated 2-manifold (BTM). This algorithm enabled the robot to traverse slopes without entangling the tether. Meanwhile, [12] examined the traversability of a rappelling rover on extreme planetary terrains by investigating the interaction between the tether and the terrain and proposed an ABIT*-based algorithm [13] to identify navigable paths. However, these studies primarily addressed the prevention of tether entanglement rather than augmenting the tether tension through tether-environment interaction.

The authors are with the Department of Mechanical and Industrial Engineering, Northeastern University, Boston, MA, USA ({xue.xing,s.yong}@northeastern.edu). This work was supported in part by an Early Career Faculty grant 80NSSC21K0071 from NASA's Space Technology Research Grants Program.

On the other hand, the study in [14] explored scenarios with winding angle constraints and solved the associated motion planning problem using the Simplicial A* algorithm [15], but it required manual determination of the winding angle requirements for each obstacle a priori, which was often suboptimal and challenging. A more recent work [16] addressed this limitation by considering a winding angle constraint for the entire tether configuration. It proposed a variant of hybrid A* [17] that utilized a modified tangent graph [18] to find a curvature-constrained path from a starting pose to a goal pose, while ensuring that the required tether winding angle was achieved by including a weighted winding cost into the total cost function. However, this altered the shortest path cost and introduced potentially inconsistent heuristics, which often led to suboptimality for the intended shortest path problem. Moreover, these works considered only a single robot with the other end of the tether being anchored, which could limit the search space.

Thus, this paper considers the scenario of a pair of robots that are both mobile (i.e., not anchored), which expands the search space and implicitly allows collaboration. Moreover, we avoid suboptimality due to the modified cost function and the inadmissible heuristic by extending the framework of *anytime A* search algorithms* [19], [20] to satisfy the required winding angle constraints. Specifically, our proposed algorithm is *anytime* in the sense that an initial suboptimal solution can be found quickly with the help of a non-admissible heuristic, while progressively refining it given additional computation time until an optimal solution is found. We also prove the completeness and optimality of the proposed algorithm for finding the shortest winding-constrained paths, and demonstrate its performance via illustrative simulations.

II. Modeling and Problem Statement

Notation: $\|\cdot\|$ denotes the Euclidean norm of its argument. $|\cdot|$ denotes the absolute value of a scalar argument or the cardinality of a set. $A \backslash B$ denotes all the elements of the set A that are not in the set B. A convex set is expressed as $conv(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N)$, where $\mathbf{v}_i \in \mathbb{R}^2$ for all $i \in \{1, 2, \ldots, N\}$ denote the vertices of the convex set and are sequentially numbered anticlockwise. The distance between two position vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ is defined as:

$$d(\mathbf{r}_1, \mathbf{r}_2) = \|\mathbf{r}_1 - \mathbf{r}_2\|. \tag{2}$$

We consider an environment $\mathcal{W} \subseteq \mathbb{R}^2$ consisting of a two-dimensional terrain with polygonal obsta-

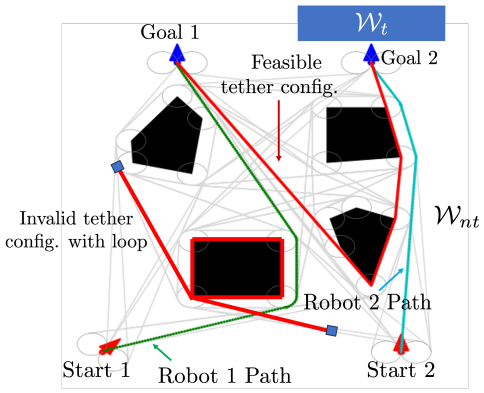


Fig. 1: Illustration of feasible and infeasible tether configurations.

cles $\mathcal{O}_k = conv\left(\{\mathbf{r}_{ok}^1, \mathbf{r}_{ok}^2, \ldots, \mathbf{r}_{ok}^{N_{v,ok}}\}\right)$, for all $k \in I_o = \{1, 2, \ldots, N_o\}$, where $\mathbf{r}_{ok}^\dagger = [x_{ok}^\dagger, y_{ok}^\dagger]^T$ for $\dagger \in \{1, 2, \ldots, N_{v,ok}\}$ are the vertices of the $\mathcal{O}_k$ and $N_{v,ok}$ denotes the total number of vertices on $\mathcal{O}_k$. The boundary of $\mathcal{O}_k$ is denoted by $\partial\mathcal{O}_k$.

Further, we consider two ground robots whose pose (i.e., position and heading) is given by $\mathbf{p} = [x, y]^T \in \mathbb{R}^2$ and θ. Both robots have a tether attached on the top connecting each other. The motion of the robots is modeled using the unicycle kinematics as follows:

$$\dot{\mathbf{r}} = \begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \omega \end{bmatrix}, \tag{3}$$

where v is a constant speed and ω is the angular speed of the robot. The angular speed is assumed to be bounded as $\omega \leq \bar{\omega}$. This imposes a curvature constraint on the path that can be traversed by the robot. The minimum turning radius possible for the robot becomes $\underline{\rho}_{turn} = \frac{v}{\bar{\omega}}$.

Inspired by [21] and [22], the boundaries $\partial\mathcal{O}_k$ are inflated by size of $\rho_{\bar{o}}$ ($> \max(\rho_r, \underline{\rho}_{turn})$) to account for safety, agent size and minimum turning radius. The inflated obstacles are denoted by $\bar{\mathcal{O}}_k$, and are given as (cf. Fig. 1): $\bar{\mathcal{O}}_k = \mathcal{O}_k \bigoplus \mathcal{B}(\rho_{\bar{o}})$, where $\bigoplus$ denotes the Minkowski sum of the sets and $\mathcal{B}(\rho_{\bar{o}})$ denotes a ball of radius $\rho_{\bar{o}}$ centered at the origin.

The following assumptions and definitions are made:

***Assumption** 1:* The length of the tether connecting the two robots is variable and finite, satisfying

$$l(\mathcal{T}) \leq l_{max}, \tag{4}$$

and the tether remains taut at a fixed height above ground during the robots' movements.

***Assumption** 2:* The robots are unable or not allowed to physically cross the tether, whereas tether contact with itself is allowed.

***Definition** 1 (Path):* A path $\mathcal{P}$ is defined as a union of path segments, $\mathcal{P} = \bigcup_{i=1}^{N_{ps}} \widetilde{\mathbf{r}_i\mathbf{r}_{i+1}}$, where $\widetilde{\mathbf{r}_i\mathbf{r}_{i+1}}$ denotes either a straight line or a circular arc segment connecting the points $\mathbf{r}_i$ and $\mathbf{r}_{i+1}$ for all $i \in \{1, 2, ..., N_{ps}\}$, where N_{ps} is the total number of path segments on the given path.

The length of a path is defined as: $l(\mathcal{P}) = \sum_{i=1}^{N_{ps}} l(\widetilde{\mathbf{r}_i\mathbf{r}_{i+1}})$, where $l(\widetilde{\mathbf{r}_i\mathbf{r}_{i+1}})$ is equal to the distance $d(\mathbf{r}_i, \mathbf{r}_{i+1})$ if the path segment is a straight line or is equal to the length of the arc if the path segment is a circular arc.

***Definition** 2 (Tether Configuration):* A tether configuration that is achieved by both robots moving on their paths $\mathcal{P}$ is defined as $\mathcal{T}(\mathcal{P}) := \bigcup_{j=1}^{N_a-1} \overline{\mathbf{a}_j\mathbf{a}_{j+1}}$ with intermediate tether segments $\overline{\mathbf{a}_j\mathbf{a}_{j+1}}$ defined similarly to path segments in Definition 1, where $\mathbf{a}_j$ is the j^{th} anchor point along the tether and N_a is the total number of anchor points on the tether.

Under Assumption 2, only a subset of all potential tether configurations is feasible for the robots to achieve, and complications related to entanglement can be prevented. Formally, we define the following:

***Definition** 3 (Feasible Tether Configuration (FTC)):* A tether configuration $\mathcal{T}(\mathcal{P})$ is considered as a feasible tether configuration if it can be achieved by the robots without having to cross the tether and obstacles.

An example of a feasible tether configuration is shown in Fig. 1. Note that a tether configuration is still feasible if the tether winds around an obstacle and only makes contact with the existing tether at a single point or a line. In this case, the tether touches itself but does not cross itself, which is still considered feasible. However, the tether loop on the left of Fig. 1 is infeasible because the tether crosses itself, causing entanglement.

At each anchor point $\mathbf{a}_j = [x_{aj}, y_{aj}]^T$ on the tether, we define the winding angle ϕ_j as the angle by which the tether bends at that anchor point. If the tether segments on either side of the given anchor point are straight lines, then the winding angle ϕ_j is defined as:

$$\phi_j = \begin{cases} \cos^{-1}\left(\frac{(\mathbf{a}_j - \mathbf{a}_{j-1})^T(\mathbf{a}_{j+1} - \mathbf{a}_j)}{|(\mathbf{a}_j - \mathbf{a}_{j-1})||(\mathbf{a}_{j+1} - \mathbf{a}_j)|}\right), & \text{if } j \in [1, N_a - 1]; \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

The total winding angle of a tether configuration $\mathcal{T}(\mathcal{P})$ is defined as follows.

***Definition** 4 (Total winding angle):* The total winding angle of a given tether configuration $\mathcal{T}$ is given by: $\phi(\mathcal{T}) = \sum_{j=1}^{N_a-1} |\phi_j|$.

The environment $\mathcal{W}$ is partitioned into two distinct regions: 1) a plateau where no additional traction is required for the robots to move [16], shown in white in Fig. 1 and denoted by $\mathcal{W}_{nt}$, and 2) an exploration region where additional traction is required for the robots to remain stable and not slide, shown in blue as a rectangle in Fig. 1 and denoted by $\mathcal{W}_t$, which can be a steep sloped terrain in practice. The tether is wrapped around

obstacles to generate a tension through friction, via the capstan effect, to support the second robot when it needs to go down the steep sloped terrain. The friction provided by the tether is a monotonically increasing function of the winding angle of the tether around the obstacle surfaces [23], given by the capstan equation:

$$f = (F_0 + \mu_{s0} m_1 g) e^{\left(\sum_{j=1}^{N_a - 1} \mu_{aj} |\phi_j|\right)}, \tag{6}$$

where F_0 is the maximum force that the robot on the plateau could provide and μ_{aj} is the friction coefficient at each obstacle vertex where the tether anchor a_j lies. Thus, to ensure a sufficient amount of friction is generated by the tether, the total winding angle on the resulting tether configuration $\mathcal{T}$ has to satisfy a desired winding angle constraint corresponding to the desired friction:

$$\phi(\mathcal{T}) \geq \phi_{des}. \tag{7}$$

Next, we describe the construction of the $\mathcal{C}^1$-tangent graph, which will be used later for motion planning, where the key idea is that paths planned based on the tangent graph are, by design, feasible paths that satisfy the curvature constraints [18].

***Definition** 5 (Tangent Graph):* For an obstacle environment with convex polygonal obstacles $\mathcal{O}_k$, the construction of the $\mathcal{C}^1$-tangent graph denoted as $\mathcal{G}_t$ involves connecting the enlarged obstacles $\bar{\mathcal{O}}_k$ as well as two circles of radius $\rho_{\bar{o}}$ on each side of the start and goal states (cf. Fig. 2) with their common tangents; then, the points at which these tangents touch the boundaries $\partial\bar{\mathcal{O}}_k$ serve as the tangent nodes on $\mathcal{G}_t$ [16].

***Definition** 6 (Tangent Node):* A tangent node $\mathbf{v}_t$ of a given $\mathcal{C}^1$-Tangent graph $\mathcal{G}_t$ is either 1) a point on the obstacle boundary $\partial\bar{\mathcal{O}}_k$, for some $k \in I_o$, where a tangent from another obstacle touches the boundary $\partial\bar{\mathcal{O}}_k$, or 2) an end point of the straight-line segments of the boundary $\partial\bar{\mathcal{O}}_k$ [16].

***Definition** 7 (Neighbor Nodes and Transit Points):* On the $\mathcal{C}^1$-tangent graph $\mathcal{G}_t$, the neighbors of a given tangent node $\mathbf{v}_t$ that lies at the vertex † of an obstacle $\mathcal{O}_k$ is a set of all the tangent nodes that are connected to the current tangent node by a transit point. A transit point is an intermediate point on the same arc as the current tangent node that forms a tangent line with the neighbor tangent nodes. To move from the current node to any of its neighboring nodes (depending on the robot orientation), the robot needs to move past respective transit points on the same arcs first and then get to the neighboring nodes.

Then, our problem of interest is as follows.

***Problem** 1:* Given the environment $\mathcal{W}$, and the start pose $\mathbf{v}_s$ and the goal pose $\mathbf{v}_g$ of the robot pair in the plateau region $\mathcal{W}_{nt} \times \mathcal{W}_{nt}$, the objective is to find the shortest paths $\mathcal{P}$ for the robot pair to move from start pose $\mathbf{v}_s$ to goal pose $\mathbf{v}_g$ such that: 1) the robots' paths satisfy the curvature constraint, 2) on the resulting paths $\mathcal{P}$, the robots avoid the obstacles, i.e., $\mathcal{P} \not\subset \mathcal{O}' \triangleq \bigcup_{k \in I_o} \{\mathcal{O}_k \bigoplus \mathcal{B}(\rho_{safe})\}$ with $\rho_{safe} > \rho_r$, 3) the resulting tether configuration $\mathcal{T}(\mathcal{P})$ is a feasible tether configuration, and 4) the winding angle constraints in Eq. (7) on the tether are satisfied.

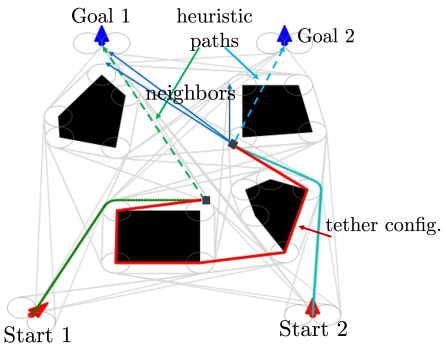


Fig. 2: Elements of $\mathcal{C}^1$-Tangent Graph and illustration of the heuristic paths for each robot.

III. Winding-Constrained Motion Planning for a Pair of Tethered Robots

In this section, we describe our path planning algorithm that combines the ideas of the winding-constrained tangent-based A* in [16] with the anytime A* approach in [19], [20] to solve Problem 1. The winding angle constraints are appropriately incorporated into the heuristic costs to expand nodes that steer the robot pair towards winding around the obstacles with its tether to satisfy Eq. (7). This is a greedy search that can generate suboptimal solutions with significantly less computational cost compared to the standard A* approach. After the suboptimal solution is generated, the algorithm will not stop, unless interrupted (hence, possessing the *anytime* property), but will continue running to find better solutions with less cost. Another characteristic is that even after it has found the optimal solution, it still keeps running until the open set is empty, which allows us to certify that the last found pair of paths is optimal.

In our proposed winding-constrained tangent-based anytime hybrid A* (WiTAH A*) algorithm, the augmented nodes of the search graph are defined as pairs of poses in $\mathcal{G}_t$ and their associated tether configurations $\mathcal{T}$. We seek to explore the poses in $\mathcal{G}_t$ that will give us the shortest paths whose associated tether configuration $\mathcal{T}$ satisfies the winding angle constraint in Eq. (7).

A. Path and Heuristic Costs

To find the shortest paths, as in anytime (weighted) A* algorithms [19], [20], we consider paths that minimize the total cost consisting of the path cost to reach a node $g(\mathsf{v}_\iota)$ and a heuristic cost to reach the goal $h(\mathsf{v}_\iota, \mathcal{T}, \omega)$,

$$f(\mathsf{v}_\iota, \mathcal{T}, \omega) = g(\mathsf{v}_\iota) + h(\mathsf{v}_\iota, \mathcal{T}, \omega), \tag{8}$$

where $\omega \geq 0$ is a heuristic weight to bias the algorithm towards generating suboptimal solutions quickly. In addition, we define the cost metrics g and h as follows:

1) Cost to reach a node: The cost already incurred by a robot pair to reach a pose $\mathsf{v}_\iota = (\mathsf{v}_{\iota 1}, \mathsf{v}_{\iota 2})$ from the start pose $\mathsf{v}_s = (\mathsf{v}_{s1}, \mathsf{v}_{s2})$ is defined as the total length of the paths of the robot pair from the start v_s to v_ι:

$$g(\mathsf{v}_\iota) = l(\mathsf{v}_\iota.\mathcal{P}_s), \tag{9}$$

where $\mathsf{v}_\iota.\mathcal{P}_s$ are Dubins paths of the robot pair from v_s to v_ι (that satisfy curvature constraints by design [18]). Note that this cost does not directly depend on the tether configuration since we seek to minimize the path lengths only, in contrast to WiTHy A* [16] that minimizes a weighted sum of the path length and winding angle cost.

2) Heuristic cost to reach the goal: Next, we define the heuristic cost as the combination of a heuristic path cost and a weighted heuristic winding cost:

$$h(\mathsf{v}_\iota, \mathcal{T}, \omega) = h_p(\mathsf{v}_\iota) + \omega h_w(\mathsf{v}_\iota, \mathcal{T}), \tag{10}$$

where the heuristic path cost (to go) is defined as the Euclidean distance between v_ι and the goal pose v_g, i.e.,

$$h_p(\mathsf{v}_\iota) = \|\mathsf{v}_\iota - \mathsf{v}_g\|_2, \tag{11}$$

and the heuristic winding cost is defined as the "penalty" for not satisfying the desired winding angle constraint ϕ_{des} in (7):

$$h_w(\mathsf{v}_\iota, \mathcal{T}) = \max(\phi_{des} - \phi(\mathcal{T}), 0), \tag{12}$$

with $\omega \geq 0$ being a winding heuristic weight and $\phi(\mathcal{T})$ being the total winding angle for the tether configuration $\mathcal{T}$ (cf. Definition 4). Note that as in anytime A* search algorithms [19], [20], the weighted heuristic cost need not be admissible when $\omega > 0$. In our approach, we choose $h_p(\mathsf{v}_\iota) = h(\mathsf{v}_\iota, \mathcal{T}, 0)$ to be admissible such that $f(\mathsf{v}_\iota, \mathcal{T}, 0)$ can be used as an upper bound for pruning.

B. The WiTAH A* Algorithm

Our proposed algorithm is described in Algorithm 1. The main idea of the WiTAH A* algorithm is to expand the reachable tangent nodes of the underlying $\mathcal{C}^1$-tangent graph for both robots by using the costs based on the path length and the inadmissible tether winding heuristic defined earlier to compare the augmented nodes, such that the first suboptimal solution is quickly generated. Then, given more time (in an anytime fashion), the algorithm leverages the suboptimal solution as an upper bound to prune the exploration while continuing to explore the graph until the optimal solution is found and certified to be optimal (when the open set becomes empty). In all the returned anytime solutions (suboptimal or optimal), the goal pose of both robots is reached, the winding angle constraint in (7) is satisfied and tether feasibility is ensured (cf. Definition 3).

The inputs for Algorithm 1 are the start and end poses of the robot pair, their initial tether configuration, map information of the obstacle locations, a winding heuristic weight ω and a weight reduction step γ, where ω and γ are tuning parameters. The *Open* set of pose-tether pairs to be explored, initialized with $(\mathsf{v}_s = (\mathsf{v}_{s1}, \mathsf{v}_{s2}), \mathcal{T}_s)$ for Robots 1 and 2, is stored as a priority queue that sorts the pose-tether pairs based on their total costs. The *Closed* set is used to store the pose-tether pairs already explored. The *incumbent* solution is the current best (but potentially suboptimal) solution to the goal pose v_g that satisfies the winding angle constraint in (7), while the *incumbentCost* is its total path cost (without the heuristic component), i.e., $f(\mathsf{v}_{g_k}, \mathcal{T}_k, 0) = g(\mathsf{v}_{g_k})$ (achieved for the k-th time), which is used as an upper bound for pruning. Its initial value is set to ∞.

The main section of the algorithm is a while loop, and

Algorithm 1: WiTAH A* Algorithm

Input: $\mathsf{v}_s, \mathsf{v}_g, \mathcal{T}_s, \bar{\mathcal{O}}, \mathcal{G}_t, \omega, \gamma$
Output: $\mathcal{P}$

Function WiTAH-A*$(\mathsf{v}_s, \mathsf{v}_g, \mathcal{T}_s, \bar{\mathcal{O}}, \mathcal{G}_t, \omega, \gamma)$**:**
  $Closed \leftarrow \emptyset$;
  $Open \leftarrow (\mathsf{v}_s, \mathcal{T}_s)$; ▷ Priority Queue
  $incumbent \leftarrow \emptyset$;
  $incumbentCost \leftarrow \infty$;
  while $Open \neq \emptyset$ **and not** *interrupted* **do**
    $(\mathsf{v}_b, \mathcal{T}_b) \leftarrow \arg\min_{(\mathsf{v},\mathcal{T}) \in Open} f(\mathsf{v}, \mathcal{T}, \omega)$;
    $Open \leftarrow Open \setminus (\mathsf{v}_b, \mathcal{T}_b)$;
    if $f(\mathsf{v}_b, \mathcal{T}_b, 0) < incumbentCost$ **then**
      $Closed \leftarrow Closed \cup (\mathsf{v}_b, \mathcal{T}_b)$;
      $\mathcal{N}(\mathsf{v}_b) =$ `childrenOf` $(\mathsf{v}_b, \mathcal{G}_t)$;
      for *each* $\mathsf{v}_n = (\mathsf{v}_{n1}, \mathsf{v}_{n2})$ *in* $\mathcal{N}(\mathsf{v}_b)$ **do**
        $c(\mathsf{v}_b, \mathsf{v}_n) \leftarrow l(\mathcal{P}_{dub}(\mathsf{v}_b, \mathsf{v}_n))$;
        $\mathcal{T}_n \leftarrow$ `updateTetherConf`$(\mathcal{T}_b, \mathcal{P}(\mathsf{v}_b, \mathsf{v}_n))$;
        if `TetherCrossed`$(\mathcal{T}_n, \mathcal{T}_b)$ **or** $g(\mathsf{v}_b) + c(\mathsf{v}_b, \mathsf{v}_n) + h(\mathsf{v}_n, \mathcal{T}_n) \geq incumbentCost$ **or** $l(\mathcal{T}_n) \geq l_{max}$ **then**
          continue;
        if `isGoal`(v_n) **and** $\phi(\mathcal{T}_n) \geq \phi_{des}$ **then**
          $incumbent =$ `extractPath`(v_n);
          $incumbentCost \leftarrow f(\mathsf{v}_n, \mathcal{T}_n, 0)$;
          if $\omega = 0$ **then**
            return *incumbent*;
          $\omega \leftarrow \max(0, \omega - \gamma)$; ▷ Decrease ω
        else if $(\mathsf{v}_n, \mathcal{T}_n) \in Open \cup Closed$ **and** $g(\mathsf{v}_n) > g(\mathsf{v}_b) + c(\mathsf{v}_b, \mathsf{v}_n)$ **then**
          $g(\mathsf{v}_n) \leftarrow g(\mathsf{v}_b) + c(\mathsf{v}_b, \mathsf{v}_n)$;
          if $(\mathsf{v}_n, \mathcal{T}_n) \in Closed$ **then**
            $Open \leftarrow Open \cup (\mathsf{v}_n, \mathcal{T}_n)$;
            $Closed \leftarrow Closed \setminus (\mathsf{v}_n, \mathcal{T}_n)$;
        else if $(\mathsf{v}_n, \mathcal{T}_n) \notin Open \cup Closed$ **then**
          $g(\mathsf{v}_n) \leftarrow g(\mathsf{v}_b) + c(\mathsf{v}_b, \mathsf{v}_n)$;
          $Open \leftarrow Open \cup (\mathsf{v}_n, \mathcal{T}_n)$;
  return *incumbent*

for every iteration the robot pair node with the smallest total cost is removed from the *Open* set. If the *Open* set is already empty or the algorithm is interrupted, the algorithm will return the current suboptimal solution.

At each iteration with an augmented node $(\mathsf{v}_b, \mathcal{T}_b)$, if a finite *incumbent* solution exists, Line 9 prunes those that are larger than the *incumbentCost* (upper bound). Specifically, if the potential path cost $f(\mathsf{v}_b, \mathcal{T}_b, 0)$ for v_b to reach the goal from the start is already larger than the current suboptimal *incumbent* solution, then this $f(\mathsf{v}_b, \mathcal{T}_b, 0)$ cannot improve on the *incumbent* solution and is thus abandoned. This is a crucial step in the anytime heuristic search algorithm to avoid the explosion of the *Open* set due to unnecessary node expansions.

Once v_b is validated, we add it to the *Closed* set and find its neighbors using the function `childrenOf`. For each neighbor (augmented) node, we then check in Lines 13–16 if the potential path cost from start to the goal is larger than the *incumbentCost* or if the tether configuration is infeasible (via the function `TetherCrossed` and maximum tether length check), and prune the invalid neighbor nodes. Next, if the neighbor node improves on the *incumbent* solution and is feasible, we will check whether the robot pair has reached the goal pose and the winding angle constraint in (7) is satisfied. If so, this augmented node will be stored as the new *incumbent* solution along with its path cost *incumbentCost*, and the weight ω is reduced by a step γ. Otherwise, we will proceed with updating the path costs of the previously generated augmented nodes (in the *Open*∪*Closed* sets) if the new neighbor node reduces the previously stored path costs and returning them to the *Open* set (and removed from *Closed*) in Lines 23–27. On the other hand, if the neighbor node is never generated (not in *Open*∪*Closed*), Lines 29–30 store its path cost and similarly append it to the *Open* set. It is worth emphasizing that in WiTAH A*, we consider augmented nodes consisting of pose-tether pairs, which differentiate among the same robot poses that have different tether configurations.

Finally, the algorithm returns the best (suboptimal) *incumbent* paths, which may be $\emptyset$ if a solution does not exist or if the algorithm is interrupted too early, i.e., before a suboptimal solution could be found.

C. Algorithm Analysis

In this section, we prove the completeness and optimality of the WiTAH A* algorithm.

1) Completeness: We begin by proving the completeness of Algorithm 1, which is a generalization of the completeness proof for WiTHy A* in [16] to the case with a pair of robots (independently of γ_{wc} in [16]).

Theorem 1: Suppose Assumption 1 and 2 hold. For any environment with a finite number of polygonal obstacles $\mathcal{O} = \{\mathcal{O}_k\}_{k\in I_o}$, if there exist paths from the initial pose v_s to the goal pose v_g that satisfy the conditions 1–4 in Problem 1, then the WiTAH A* algorithm in Algorithm 1 will find such paths in a finite number of iterations or declare that no such paths exist (*incumbent* is empty).

Proof: The proof generalizes that of [16, Theorem 1] and follows a similar logic. Specifically, note that our search space is the augmentation of the tangent nodes of the robot pair on the $\mathcal{C}^1$-tangent graph $\mathcal{G}_t$ with their associated tether configuration. Since we have a finite number of obstacles in the environment, we have a finite number of tangent nodes. Moreover, under Assumption 1 and 2, there cannot be an infinite number of tether configurations and paths because they would eventually end up crossing each other or the tether would exceed its maximum length. Hence, by design, the search space of the algorithm is finite. Since our algorithm does not terminate until a solution with the desired winding angle is found if a solution exists and the algorithm is not interrupted too early before a suboptimal solution can be generated, the algorithm will find feasible winding-constrained paths in a finite number of iterations or declare that no such paths exist (with an empty *incumbent*); thus, the algorithm is complete. ■

2) Optimality: Next, we prove the optimality of the WiTAH A* algorithm if given sufficient time.

Theorem 2: Suppose Assumption 2 holds. For any environment with a finite number of polygonal obstacles $\mathcal{O} = \{\mathcal{O}_k\}_{k\in I_o}$, the algorithm always terminates (in a finite number of iterations) and its final solution is optimal.

Proof: Suppose that the algorithm terminates before finding an optimal solution with minimum path cost f^*. The sequence of upper bounds used during execution of the algorithm is $b_0, b_1, \ldots, b_k$, where $b_0 = \infty$ is the initial *incumbentCost* before any solution is found, b_1 is the cost of the first *incumbent* solution found, and b_k is the cost of the last *incumbent* solution found. Due to the pruning step of the algorithm in Line 9, it follows that

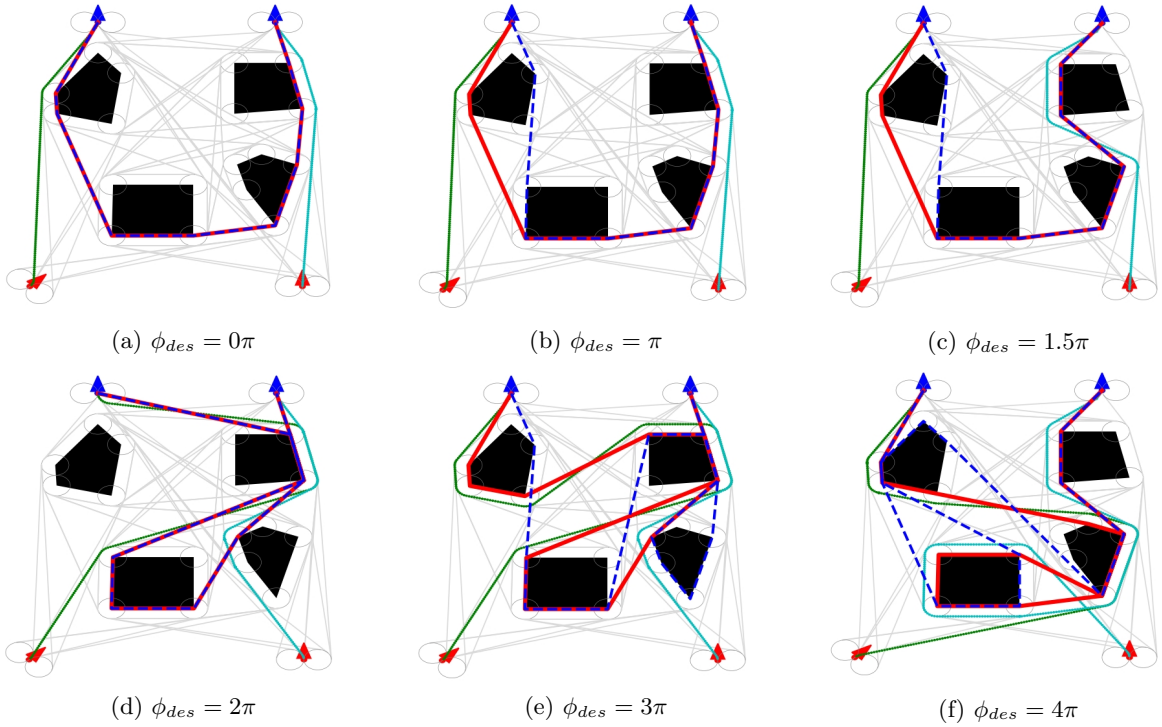
$$b_0 > b_1 > \cdots > b_k > f^*,$$

where the last inequality holds under the assumption that the algorithm terminates at a finite k with a suboptimal solution before finding the optimal solution. Now consider optimal paths leading from the initial state to a goal state with winding angle constraint satisfied. Under the assumption that these optimal paths were not found, there must be some node v_b along these paths that was not explored. That is only possible if $f(\mathsf{v}_b, \mathcal{T}_b, 0) = g(\mathsf{v}_b) + h_p(\mathsf{v}_b) \geq b_k$.

However, by the admissibility of h_p (cf. end of Section III-A), we know that

$$g(\mathsf{v}_b) + h_p(\mathsf{v}_b) \leq f^* < b_i$$

for all i, including $i = k$. From this contradiction, it follows that the algorithm cannot terminate before an optimal solution is found. ■



(a) $\phi_{des} = 0\pi$ (b) $\phi_{des} = \pi$ (c) $\phi_{des} = 1.5\pi$

(d) $\phi_{des} = 2\pi$ (e) $\phi_{des} = 3\pi$ (f) $\phi_{des} = 4\pi$

Fig. 3: Comparison of the fast suboptimal (first) solutions and the shortest (final) paths for different ϕ_{des} (in radians) using the proposed WiTAH A* algorithm. (Red arrows represent the start pose and the blue arrows represent the goal pose. The gray lines depict the edges of the tangent graph. The green path is the optimal path that Robot 1 takes and the light blue path is the optimal path that Robot 2 takes. The blue dashed line is the tether configuration of the suboptimal solution and the red line is the tether configuration of the optimal solution).

Note that by design and similarly to anytime A* search algorithms, e.g., [19], [20], the choices of ω and γ do not affect the optimality of our WiTAH A* algorithm, but they may affect when the first suboptimal solution is found and when the algorithm terminates.

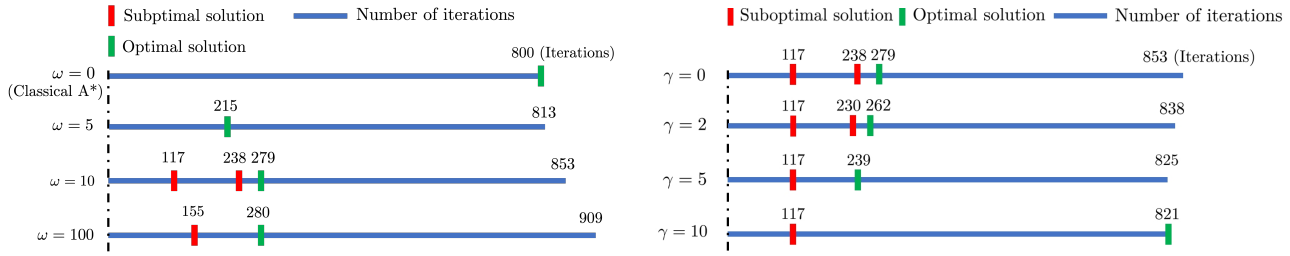
IV. Experiment Results and Discussion

In this section, we studied the performance of the proposed algorithm via simulation experiments. We considered an environment with 4 obstacles defined as: $\mathcal{O}_1 = conv([6,5]^T, [12,5]^T, [12,10]^T, [6.1,10]^T)$, $\mathcal{O}_2 = conv([15,17]^T, [20,17.5]^T, [19,22]^T, [15,22]^T)$, $\mathcal{O}_3 = conv([15.9,9.5]^T, [18,6]^T, [19.6,12]^T, [17,13]^T, [15.2,12]^T)$, $\mathcal{O}_4 = conv([2,17]^T, [6,16]^T, [6.7,21]^T, [5,23]^T, [1.9,19]^T)$. The robot pair started at $\mathsf{v}_{s1} = [0,0,\frac{\pi}{4}]^T, \mathsf{v}_{s2} = [20,0,\frac{\pi}{2}]^T$ and the goal pose was $\mathsf{v}_{g1} = [5,26,\frac{\pi}{3}]^T, \mathsf{v}_{g2} = [18,26,\frac{\pi}{2}]^T$. l_{max} was set to 200. The paths obtained by our WiTAH A* algorithm for various choices of desired winding angle ϕ_{des} with $\omega = 10$ and $\gamma = 10$ are shown in Fig. 3 and the corresponding animations can be found at https://tinyurl.com/3kkn933w. The corresponding path lengths and numbers of iterations to obtain the fast-suboptimal and slow-optimal solutions using Algorithm 1 are given in Table I, which illustrate a trade-off between optimality and computational cost (in terms of number of iterations).

We observed that the suboptimal solutions were much faster to generate, as desired, and the suboptimal path cost became an upper bound to prune the nodes in the *Open* set. The optimal paths could also be found relatively quickly, as indicated by the asterisk (*) in the last column, where the first solution was already optimal. However, certifying its optimality usually incurred a large computational cost (final column indicates the total number of iterations from algorithm start to the final certification step) as the *Open* set, while finite, must be exhaustively evaluated before the solution can be certified as being optimal. When no winding angle constraint was imposed ($\phi_{des} = 0$), the algorithm reduced to the classical (tangent-based) A* search. On the other hand, when ϕ_{des} was non-zero, the winding heuristic biased the search towards the tether configurations satisfying the winding angle constraint.

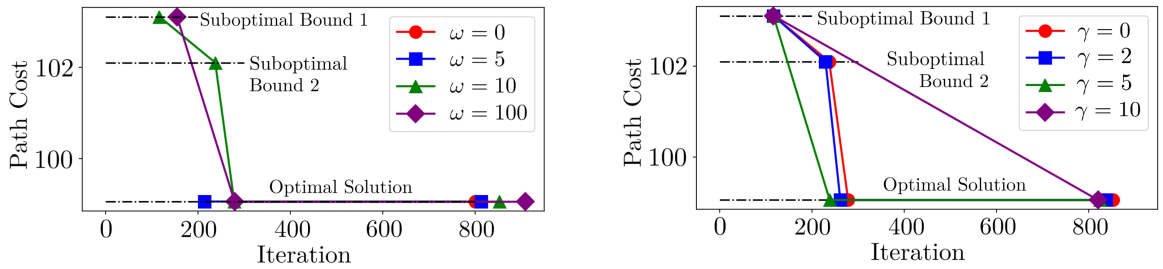
TABLE I: Comparison of the first suboptimal solution and the optimal solution with $\omega = 10$ and $\gamma = 10$ under different winding angle constraints. * indicates the cases where the first suboptimal solution is already optimal.

ϕ_{des}	$\mathcal{P}_{sub}$		$\mathcal{P}_{opt}$	
	$l(\mathcal{P})$	Iteration	$l(\mathcal{P})$	Iteration
0π	53.90	7	53.90	7*
π	54.90	4	53.90	10
1.5π	59.68	17	58.67	53
2π	81.73	223	81.73	430*
3π	103.11	117	99.05	821
4π	126.96	623	124.83	2072



(a) $\omega \in \{0, 5, 10, 100\}; \gamma = 0$ (b) $\omega = 10; \gamma \in \{0, 2, 5, 10\}$

Fig. 4: (a) Impact of varying ω with fixed $\gamma = 0$ and $\phi_{des} = 3\pi$ and (b) impact of varying γ with fixed $\omega = 10$ and $\phi_{des} = 3\pi$.



(a) $\omega \in \{0, 5, 10, 100\}; \gamma = 0$ (b) $\omega = 10; \gamma \in \{0, 2, 5, 10\}$

Fig. 5: (a) Impact of varying ω on the suboptimal bounds with fixed $\gamma = 0$ and $\phi_{des} = 3\pi$ and (b) impact of varying γ on the suboptimal bounds with fixed $\omega = 10$ and $\phi_{des} = 3\pi$.

Next, we investigated the effect of varying the weight ω (with reduction step chosen as $\gamma = \omega$) on the resulting path lengths and numbers of iterations for their suboptimal-fast and optimal-slow solutions obtained for the case when $\phi_{des} = 3\pi$ rad. As can be observed in Table II, the choice of $\omega = 5$ led to an optimal solution with the fewest number of iterations, which was a reduction of 74.5% when compared to the classical A* (i.e., when $\omega = 0$). Moreover, we observed that when the ω was large enough, the winding heuristic played a more dominant role than the path cost in the priority queue, and in such cases, the first suboptimal solutions could be generated much faster than the optimal solution, with a reduction in computation time between 82.3% and 85.7% when $\omega \geq 10$. In contrast, when ω was chosen to be small, the path cost played a more dominant role and the first suboptimal solution often required a larger number of iterations but the optimal solution was obtained earlier. When $\omega = 0$, the whole algorithm reduced to the classical A* algorithm and could always give the optimal solution if it existed, but at a higher computational cost.

Besides the inflated heuristic coefficient ω with non-zero reduction steps, we also evaluated the effect of fixing ω (i.e., when $\gamma = 0$) on algorithm performance. In this scenario, the reduction step was set to zero so the ω remained constant even after the suboptimal incumbent solution had been found. Fig. 4(a) shows that our proposed algorithm has robust performance with selections of $\omega \geq 5$, consistently finding the optimal solution faster than the classical A* (with $\omega = 0$). Moreover, Fig. 5(a) plots the trends of the *incumbentCost* as the upper bounds, where it can be clearly seen that *incumbentCost* monotonically decreases to the optimal value in fewer iterations for all $\omega \neq 0$ than the classical A* algorithm.

TABLE II: Path data and associated computational cost for various ω (with $\gamma = \omega$) under the desired winding angle $\phi_{des} = 3\pi$. * indicates the cases where the first suboptimal solution is already optimal.

ω	$\mathcal{P}_{sub}$		$\mathcal{P}_{opt}$	
	$l(\mathcal{P})$	Iteration	$l(\mathcal{P})$	Iteration
100	103.11	155	99.05	877
10	103.11	117	99.05	821
5	99.05	215	99.05	805*
1	99.05	641	99.05	801*
0	99.05	800	99.05	800*

Finally, we studied the effect of varying the weight reduction parameter γ with a fixed $\omega = 10$. The results from Figs. 4(b) and 5(b) show that $\gamma = 5$ yielded the best performance in terms of the number of iterations (corresponding to computation time) needed to obtain the optimal solution. The $\gamma = 0$ and $\gamma = 2$ cases shared similar performance and generated two suboptimal solutions before finding the optimal one, while the $\gamma = 10$ case resulted in the worst performance in terms of iterations required to find the optimal solution. In the latter case, ω was reduced to 0 after the first suboptimal solution was found and the winding heuristic term disappeared; thus, the rest of the algorithm ran as a classical A* without the guidance of the winding angle constraint,

resulting in poor performance. In practice, the tuning of γ and ω may be problem-dependent and difficult to determine a priori. Since a fixed-weight WiTAH A* algorithm (i.e., with $\gamma = 0$) is generally more robust, easier to implement, and produces comparable results to other weight reduction steps, as was observed in several examples for the Anytime Weighted A* in [19], choosing γ as 0 is often a good starting point.

In summary, from the experiments, it is clear that the proposed WiTAH A* algorithm is superior in performance to the classical A* algorithm as well as our previous WiTHy A* algorithm [16]. In contrast to WiTHy A*, WiTAH A* does not need careful selection and fine-tuning of parameters to obtain the optimal paths, as was done by WiTHy A*, as illustrated in [16, Table II]. Moreover, the WiTAH A* algorithm also does not require the heuristic cost function to be admissible for certifying optimality. However, as stated in previous anytime A* search studies in [19], [20], the number of iterations required to fully consider all nodes in the *Open* set with $\omega \neq 0$ is slightly larger than the classical A*, since the algorithm will explore the (augmented) nodes that the classical A* do not due to the heuristic winding bias.

V. Conclusions

This paper introduced a winding-constrained anytime hybrid A* motion planning algorithm for a pair of tethered robots. The proposed algorithm integrated a modified tangent graph with an anytime A* framework to compute the shortest feasible paths for a pair of tethered curvature-constrained robots, while ensuring compliance with winding angle constraints imposed on the tether configuration. The algorithm's effectiveness was assessed through simulation experiments, which demonstrated that it reliably identified shortest paths for both robots to satisfy the desired winding angle constraints, assuming such paths existed between the specified start and goal poses within a given obstacle environment. Further, the algorithm was proven to be complete and optimal. Future work will consider winding-constrained motion planning for tethered robots on sloped and uncertain terrains.

References

[1] P. McGarey, F. Pomerleau, and T. D. Barfoot, "System design of a tethered robotic explorer (TReX) for 3D mapping of steep terrain and harsh environments," in *Field and Service Robotics: Results of the 10th International Conference*. Springer, 2016, pp. 267–281.

[2] P. McGarey, D. Yoon, T. Tang, F. Pomerleau, and T. D. Barfoot, "Developing and deploying a tethered robot to map extremely steep terrain," *Journal of Field Robotics*, vol. 35, no. 8, pp. 1327–1341, 2018.

[3] I. A. Nesnas, J. B. Matthews, P. Abad-Manterola, J. W. Burdick, J. A. Edlund, J. C. Morrison, R. D. Peters, M. M. Tanner, R. N. Miyake, B. S. Solish *et al.*, "Axel and DuAxel rovers for the sustainable exploration of extreme terrains," *Journal of Field Robotics*, vol. 29, no. 4, pp. 663–685, 2012.

[4] P. Abad-Manterola, I. A. Nesnas, and J. W. Burdick, "Motion planning on steep terrain for the tethered axel rover," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 4188–4195.

[5] M. M. Tanner, J. W. Burdick, and I. A. Nesnas, "Online motion planning for tethered robots in extreme terrain," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 5557–5564.

[6] S. Kim, S. Bhattacharya, and V. Kumar, "Path planning for a tethered mobile robot," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1132–1139.

[7] S. Kim and M. Likhachev, "Path planning for a tethered robot using multi-heuristic a with topology-based heuristics," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 4656–4663.

[8] O. Salzman and D. Halperin, "Optimal motion planning for a tethered robot: Efficient preprocessing for fast shortest paths queries," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4161–4166.

[9] L. Petit and A. L. Desbiens, "Tape: Tether-aware path planning for autonomous exploration of unknown 3D cavities using a tangle-compatible tethered aerial robot," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 550–10 557, 2022.

[10] X. Zhang and Q.-C. Pham, "Planning coordinated motions for tethered planar mobile robots," *Robotics and Autonomous Systems*, vol. 118, pp. 189–203, 2019.

[11] M. Cao, K. Cao, S. Yuan, T.-M. Nguyen, and L. Xie, "NEPTUNE: Nonentangling trajectory planning for multiple tethered unmanned vehicles," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2786–2804, 2023.

[12] M. Paton, M. P. Strub, T. Brown, R. J. Greene, J. Lizewski, V. Patel, J. D. Gammell, and I. A. Nesnas, "Navigation on the line: Traversability analysis and path planning for extreme-terrain rappelling rovers," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7034–7041.

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

[14] D. S. Yershov, P. Vernaza, and S. M. LaValle, "Continuous planning with winding constraints using optimal heuristic-driven front propagation," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 5551–5556.

[15] D. S. Yershov and S. M. LaValle, "Simplicial Dijkstra and A* algorithms: From graphs to continuous spaces," *Advanced Robotics*, vol. 26, no. 17, pp. 2065–2085, 2012.

[16] V. S. Chipade, R. Kumar, and S. Z. Yong, "WiTHy A*: Winding-constrained motion planning for tethered robot using Hybrid A*," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 8771–8777.

[17] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *The International Journal of Robotics Research*, vol. 29, no. 5, pp. 485–501, 2010.

[18] V. S. Chipade and D. Panagou, "Approximate time-optimal trajectories for damped double integrator in 2D obstacle environments under bounded inputs," *arXiv preprint arXiv:2007.05155*, 2020.

[19] E. A. Hansen and R. Zhou, "Anytime heuristic search," *Journal of Artificial Intelligence Research*, vol. 28, pp. 267–297, 2007.

[20] M. Likhachev, G. J. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality," *Advances in neural information processing systems*, vol. 16, 2003.

[21] R. Hegde and D. Panagou, "Multi-agent motion planning and coordination in polygonal environments using vector fields and model predictive control," in *2016 European Control Conference (ECC)*. IEEE, 2016, pp. 1856–1861.

[22] W. D. Esquivel and L. E. Chiang, "Nonholonomic path planning among obstacles subject to curvature restrictions," *Robotica*, vol. 20, no. 1, pp. 49–58, 2002.

[23] R. Kumar, V. S. Chipade, and S. Z. Yong, "Stability of tethered ground robots on extreme terrains," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 4542–4547.