2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

WiTHy A*: Winding-Constrained Motion Planning for Tethered Robot using Hybrid A*

Vishnu S. Chipade, Rahul Kumar, Sze Zheng Yong

***Abstract*— In this paper, a variant of hybrid A* is developed to find the shortest path for a curvature-constrained robot, that is tethered at its start position, such that the tether satisfies user-defined winding angle constraints. A variant of tangent graphs is used as an underlying graph for searching a path using A* in order to reduce the overall computation and define appropriate cost metrics to ensure winding angle constraints are satisfied. Conditions are provided under which the proposed algorithm is guaranteed to find a winding angle constrained path. The effectiveness and performance of the proposed algorithm are studied in simulation.**

I. Introduction

Tethered robots, e.g., TReX [1], [2], are able to navigate extreme terrains such as steep slopes on planetary missions using the tensions provided by the tethers. A successful exploration operation of this robot type on extreme terrains (steep slopes) relies on the generation of the desired amount of tension on the tether that may not always be readily available. In such cases, the tether can be wind around objects in the environment to induce the capstan effect and the resulting friction can augment the tension provided by the tether for robot navigation on steep slopes. While there is some research on motion planning for a single tethered robot [3]–[9] as well as multiple tethered robots [10], [11], the focus has been mostly on ensuring that the tether is not entangled, and not much consideration is given to the tension provided by the tether. In this paper, we study a path-planning problem for tethered robots in environments where only a finite amount of tension can be generated on a tether, and the friction between the tether and the obstacles is used to augment the total force provided by the given tether configuration for the successful operation of the tethered robots in extreme regions.

In [4], [5], the authors develop a motion planning algorithm to find a path for a tethered robot in the same homotopy class to navigate extreme terrains up and down without tether entanglement using boundary triangulated 2-manifold (BTM). While the algorithm does result in paths that will allow the robot to navigate slopes up and down without entanglement, the tether is assumed to support any amount of tension which is not always possible on planetary extreme terrains. Authors in [12] study the traversability of a rappelling rover on extreme planetary terrains by studying the tether-terrain interaction and also provide an ABIT* [13] based algorithm to recommend traversable paths.

Authors in [14] study the problem of motion planning with winding angle constraints. The winding angle constraints are specified as the total winding of the path desired around each of the obstacles. The winding angle constraints are carefully combined into a single constraint and the state space is projected onto a higher-dimensional covering space. The authors use Simplicial A* algorithm [15] to solve the winding angle constrained path planning problem on the higher dimensional covering space. However, determining the winding angle constraints around each obstacle a priori to ensure a desired amount of friction between the tether and the obstacle surfaces (induced due to the capstan effect) is very challenging.

In this paper, different from [14], we consider a single winding angle constraint on the entire tether configuration. Since the friction between the tether and the obstacle surfaces is characterized by the total winding of the tether around the obstacles, the total winding angle constraint will ensure a desired amount of friction can be generated between the tether and the object surfaces. We develop a winding-constrained tangent-based Hybrid A* (WiTHy A*), a variant of the Hybrid A* search algorithm [16], that uses a variant of tangent graph [17] as an underlying graph to find an optimal curvature-constrained path from a start state to a goal state such that a user-defined amount of friction induced due to capstan effect is achieved along the tether connected to the robot. Compared to the existing literature, the contributions of this paper are: 1) Tangent graph-based variant of A* to find an optimal path with winding angle constraints, 2) In the absence of the winding angle constraints, the resulting tangent-based variant of hybrid A* algorithm provides a computationally efficient path planner for curvature-constrained robots to find the shortest path in polygonal obstacle-populated environments.

II. Modeling and Problem Statement

Notation: $\|\cdot\|$ denotes the Euclidean norm of its argument. $|\cdot|$ denotes the absolute value of a scalar argument and cardinality if the argument is 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 $conv(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, ..., \mathbf{v}_N)$ where

The authors are with the Department of Mechanical and Industrial Engineering, Northeastern University, Boston, MA, USA ({v.chipade,kumar.rahul4,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.

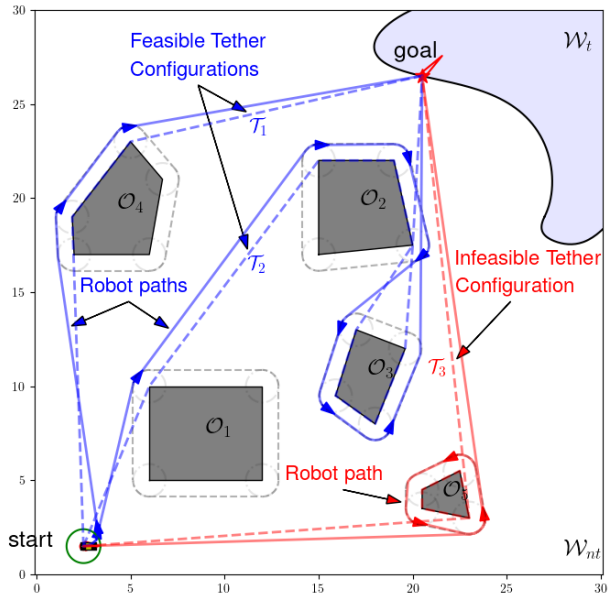


Fig. 1: Description of problem setup and illustrations of feasible and infeasible tether configurations.

$\mathbf{v}_i \in \mathbb{R}^2$ denote the vertices of the convex set and are sequentially numbered anticlockwise. 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 $\mathcal{W} \subseteq \mathbb{R}^3$ consisting of a flat terrain with polygonal shaped obstacles $\mathcal{O}_k = conv\left(\{\mathbf{r}_{ok}^1, \mathbf{r}_{ok}^2, ..., \mathbf{r}_{ok}^{N_{v,ok}}\}\right)$, for all $k \in I_o = \{1, 2, ..., N_o\}$ where $\mathbf{r}_{ok}^{\dagger} = [x_{ok}^{\dagger}, y_{ok}^{\dagger}]^T$ for $\dagger \in \{1, 2, 3, ..., N_{v,ok}\}$ are the vertices of the $\mathcal{O}_k$ and $N_{v,ok}$ denote 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 a ground robot whose position and heading are given by $\mathbf{p} = [x, y]^T \in \mathbb{R}^2$ and θ, respectively, has a circular footprint of size ρ_r, and has a tether attached on the top. The tether is anchored at the initial position of the robot. The motion of the robot is modeled using the unicycle kinematics as:

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

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 from [18] and [19], 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.

We make the following assumptions about the tether.

***Assumption** 1:* The tether connected to the robot: 1) is retractable, 2) can be ensured to stay taut during the motion of the robots.

***Assumption** 2:* The robot cannot physically cross the tether.

We define a path and tether configurations as follows.

***Definition** 1 (Path):* A path $\mathcal{P}$ is defined as a union of path segments, $\mathcal{P} = \cup_{i=1}^{N_{ps}} \widearc{\mathbf{r}_i\mathbf{r}_{i+1}}$, where $\widearc{\mathbf{r}_i\mathbf{r}_{i+1}}$ 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: $l(\mathcal{P}) = \sum_{i=1}^{N_{ps}} l(\widearc{\mathbf{r}_i\mathbf{r}_{i+1}})$, where $l(\widearc{\mathbf{r}_i\mathbf{r}_{i+1}})$ 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 the robot moving on a path $\mathcal{P}$ is defined as $\mathcal{T}(\mathcal{P}) := \cup_{j=1}^{N_a - 1} \overline{\mathbf{a}_j\mathbf{a}_{j+1}}$ 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 some of all the tether configurations are possible to achieve for the robot. Formally, we define the following.

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

The examples of feasible tether configurations are given in Figure 1. Note that a tether configuration is still feasible if the tether winds around an obstacle and only touches the existing tether at a point and then deviates away from that point, for example, $\mathcal{T}_2$ in Fig. 1. This is because the tether only touches and the robot does not have to cross the tether to achieve this configuration.

At each anchor point $\mathbf{a}_j = [x_{aj}, y_{aj}]^T$ on the tether, we define 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:

$$\phi_j = \begin{cases} \cos^{-1}\left(\frac{(\mathbf{a}_j - \mathbf{a}_{j-1})^T(\mathbf{a}_{j+1} - \mathbf{a}_j)}{|(\mathbf{a}_j - \mathbf{a}_{j-1})||(\mathbf{a}_{j+1} - \mathbf{a}_j)|}\right), & \text{if } j \in [1, N_a - 1]; \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

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 on the given tether configuration $\mathcal{T}$ is given as $\phi(\mathcal{T}) = \sum_{j=1}^{N_a - 1} |\phi_j|$.

The environment $\mathcal{W}$ is divided into two parts: 1) section where no additional traction is required for the robot to move, shown in white color in Fig. 1 and denoted by $\mathcal{W}_{nt}$, 2) section where additional traction is required for the robot to successfully move, shown in light blue color in Fig. 1 and denoted by $\mathcal{W}_t$. For the robot to successfully move from a start state to a goal state in $\mathcal{W}_t$ region, it requires a sufficient amount of traction. The tether when

wound around an obstacle provides some friction in the opposite direction of the pull by the robot. The friction provided by the tether is a function of the winding angle of the tether around the obstacle surface. Thus, to ensure a sufficient amount of friction is generated by the tether for the robot's successful motion, the total winding angle on the resulting tether configuration $\mathcal{T}$ has to satisfy:

$$\phi(\mathcal{T}) \geq \phi_{des}. \tag{5}$$

The problem addressed in this paper is formally defined as follows.

***Problem** 1:* Given the environment $\mathcal{W}$, the start state $\mathbf{r}_s$ of the robot in $\mathcal{W}_{nt}$, the goal state $\mathbf{r}_g$ in $\mathcal{W}_t \cap \mathcal{W}_{nt}$, the objective is to find the shortest path $\mathcal{P}$ for the robot to move from start state $\mathbf{r}_s$ to goal state $\mathbf{r}_g$ such that 1) the path satisfies the curvature constraint, 2) on the resulting path $\mathcal{P}$ the robot avoids the obstacles, i.e., $\mathcal{P} \not\subset \mathcal{O}' \triangleq \{\mathcal{O}_k \bigoplus \rho_{safe} \ | k \in I_o\}$, 3) the resulting tether configuration $\mathcal{T}(\mathcal{P})$ is a feasible tether configuration, and 4) the winding angle constraints in Eq. (5) on the tether are satisfied.

III. Winding Angle Constrained Motion Planning

In this section, we describe our path planning algorithm that uses the idea of $\mathcal{C}^1$-tangent graph [17] and the hybrid A* graph search approach [16] together to solve Problem 1. The idea is to use the $\mathcal{C}^1$-tangent graph defined for a given obstacle-populated environment as the underlying graph for the hybrid A* algorithm. Winding around the obstacles is added as an additional state of a given node in the search process. The winding angle constraints are appropriately incorporated into the path costs and the heuristic costs in order to expand nodes that force the winding angle toward the desired winding angle value. The details of the approach are discussed in the following subsections. We first revisit the idea of the tangent graph in the next subsection and then in the following subsection, we describe the modified version of the hybrid A* algorithm that uses this tangent graph for searching a winding angle constrained shortest path.

A. $\mathcal{C}^1$-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 $\bar{\mathcal{O}}_k$ with their common tangents; then, the points at which these tangents touch the boundaries $\partial\bar{\mathcal{O}}_k$ serve as the nodes on $\mathcal{G}_t$. These nodes called tangent nodes are formally defined as follows.

***Definition** 5 (Tangent Node):* A tangent node $\mathbf{v}_{t\ell}$ in 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 some other obstacle touches the boundary $\partial\bar{\mathcal{O}}_k$, or 2) the end points of the straight line segments of the boundary $\partial\bar{\mathcal{O}}_k$.

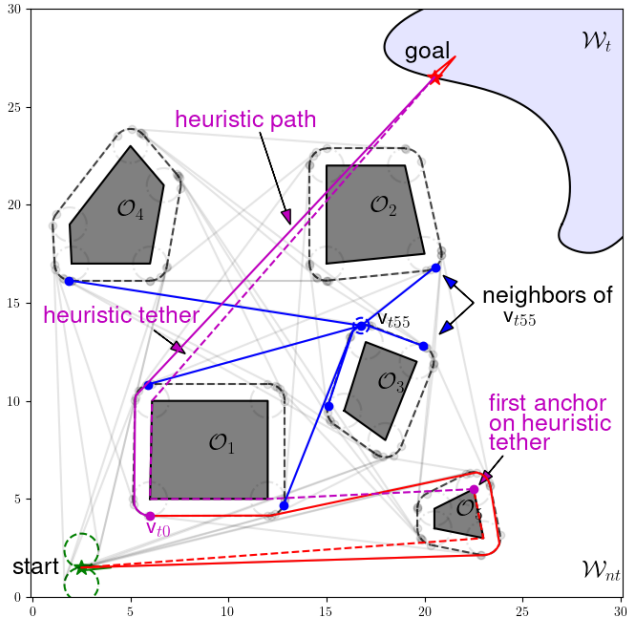


Fig. 2: Elements of $\mathcal{C}^1$-Tangent Graph and illustration of heuristic path and the heuristic tether configuration.

A tangent node is said to lie on a vertex † of an obstacle $\mathcal{O}_k$ if the point is situated on the circular segment of the boundary $\partial\bar{\mathcal{O}}_k$ around that vertex.

***Definition** 6 (Neighboring tangent nodes):* On the $\mathcal{C}^1$-tangent graph $\mathcal{G}_t$, the neighbors of a given tangent node $\mathbf{v}_{t\ell}$ that lies at vertex † of an obstacle $\mathcal{O}_k$ is a set of all the tangent nodes that are connected to the other tangent nodes on the same vertex † of $\mathcal{O}_k$ by a straight, tangent line segment (for example, see the tangent node $\mathbf{v}_{t55}$ on the vertex of the obstacle $\mathcal{O}_3$ shown in blue and its neighbors also shown in blue in Fig. 2). Let $\mathcal{N}(\mathbf{v}_{t\ell})$ denote the set of neighboring nodes of a given node $\mathbf{v}_{t\ell}$.

B. Winding-constrained tangent-based Hybrid A*

Now we describe the modified version of the hybrid A* algorithm that uses a $\mathcal{C}^1$-tangent graph $\mathcal{G}_t$ as the underlying graph for search. Before we describe the modified hybrid A* algorithm, we describe the modified definitions of the components of the A* algorithm as follows.

1) Graph Node: The tangent node in the tangent graph is used to define a graph node. In addition to the underlying tangent node $\mathsf{v}_{t\ell}$, the given node v_ι has the following additional attributes that are useful for the search algorithm.

- Underlying tangent node, $\mathsf{v}_\iota.\mathbf{v}_t = \mathbf{v}_{t\ell}$ for some ℓ;
- Parent node: the node from which the node v_ι is created, $\mathsf{v}_\iota.parent$;
- Path from the start node to the node v_ι, $\mathsf{v}_\iota.\mathcal{P}_s = \mathsf{v}_\iota.parent.\mathcal{P}_s \cup \mathcal{P}(\mathsf{v}_\iota.parent, \mathsf{v}_\iota)$;
- Tether configuration associated with the path from the start node to the node v_ι:

$$\mathsf{v}_\iota.\mathcal{T}_s = \texttt{updateTetherConf}(\mathsf{v}_\iota.parent.\mathcal{T}_s, \mathsf{v}_\iota.\mathcal{P}).$$

The function `updateTetherConf` (based on the anchor prediction algorithm in [12]) returns the updated tether configuration when the robot moves on the input path considering the given input tether configuration as the initial tether configuration;

- Comparison operator: The comparison operator to compare whether two nodes are the same or not is overloaded such that the two nodes are said to be the same if the entire history of the parents of the nodes all the way until the start node is same otherwise they are said to be different.

2) Path costs: We seek to explore nodes in $\mathcal{G}_t$ that will give us the shortest path that satisfies the winding angle constraint in Eq. (5). To do this, we first define a cost metric associated with the winding angle constraint as:

$$f_{wc}(\phi) = \max(\phi_{des} - \phi, 0). \quad (6)$$

We then define the following weighted cost metrics.

Cost to reach a node: The cost to reach a node v_ι from the start node is defined as a weighted sum of the total length of the path from the start to v_ι and the winding angle constraint cost (Eq. (6)) of the corresponding tether configuration:

$$g[\mathsf{v}_\iota] = (1 - \gamma_{wc}) * l(\mathsf{v}_\iota.\mathcal{P}_s) + \gamma_{wc} * f_{wc}(\phi(\mathsf{v}_\iota.\mathcal{T}_s)), \quad (7)$$

where $\gamma_{wc} \in [0, 1]$ is a user-defined weightage given to the winding angle constraint cost.

Heuristic cost to reach the goal: For a given node v_ι, we first define a heuristic path to goal $\mathcal{P}_{heu}(\mathsf{v}_\iota, \mathsf{v}_g)$ which is the shortest path from the node v_ι to the goal that goes around the obstacle on which the node v_ι lies but does not consider collisions with other obstacles (see the magenta path shown in Fig. 2), denoted by $\mathsf{v}_\iota.\mathcal{P}_g = \mathcal{P}_{heu}(\mathsf{v}_\iota, \mathsf{v}_g)$. The corresponding heuristic tether configuration is given as $\mathsf{v}_\iota.\mathcal{T}_g = \mathcal{T}_{\mathsf{v}_\iota}(\mathcal{P}_{heu}(\mathsf{v}_\iota, \mathsf{v}_g))$, where $\mathcal{T}_{\mathsf{v}_\iota}(\mathcal{P}_{heu}(\mathsf{v}_\iota, \mathsf{v}_g))$ is the tether configuration achieved by the robot after moving on the path $\mathcal{P}_{heu}(\mathsf{v}_\iota, \mathsf{v}_g)$ and considering the first anchor point to be the same as the last anchor point of the tether configuration $\mathsf{v}_\iota.\mathcal{T}_s$ (see Fig. 2). The winding angle at the first anchor point of this heuristic tether is the angle made by the first tether segment on the heuristic tether with the last segment of the tether configuration $\mathsf{v}_\iota.\mathcal{T}_s$.

The heuristic cost associated with the heuristic path from the node to the goal and the corresponding heuristic tether configuration is defined as:

$$h[\mathsf{v}_\iota] = (1 - \gamma_{wc}) * l(\mathsf{v}_\iota.\mathcal{P}_g) + \gamma_{wc} * f_{wc}(\phi(\mathsf{v}_\iota.\mathcal{T}_g)). \quad (8)$$

The estimate of the cost of the path going through v_ι to the goal is then obtained based on $\mathcal{P}_{est}(\mathsf{v}_\iota)$ as follows:

$$f[\mathsf{v}_\iota] = g[\mathsf{v}_\iota] + h[\mathsf{v}_\iota]. \quad (9)$$

3) The WiTHy A Algorithm:* The modified version of the hybrid A* which we call winding-constrained tangent-based hybrid A* (abbreviated as WiTHy A*) is described in Algorithm 1. The main idea in the WiTHy A* algorithm is to 1) expand the reachable tangent nodes on the vertices of the nearby obstacles in the underlying $\mathcal{C}^1$-tangent graph, 2) use costs based on the path length and the tether winding defined earlier to compare the nodes, 3) continue exploring the graph until the goal state is reached and the winding angle constraints are satisfied and the tether feasibility is ensured.

Algorithm 1: WiTHy A*: Winding-Constrained Tangent-Based Hybrid A*

Input: $\mathsf{v}_s \leftarrow \mathbf{r}_i$, $\mathsf{v}_g \leftarrow \mathbf{r}_g$, $\mathcal{O}'$, $\mathcal{G}_t$, f, g, h
Output: $\mathcal{P}$

1 **Function** `WiTHyA*` $(\mathsf{v}_s, \mathsf{v}_g, \mathcal{G}_t)$**:**
2 $\mathcal{V}_{op} \leftarrow \{\mathsf{v}_s\}$;
3 $\mathcal{V}_{cl} \leftarrow \{\}$;
4 **while** $\mathcal{V}_{op}$ *is not empty* **do**
5 $\mathsf{v}_b = \arg\min_{v \in \mathcal{V}_{op}} f[v]$;
6 $\mathcal{P}_{dub}(\mathsf{v}_b, \mathsf{v}_g) =$ `dubinsPath`$(\mathsf{v}_b, \mathsf{v}_g)$
7 **if** $\mathcal{P}_{dub}(\mathsf{v}_b, \mathsf{v}_g) \subset \mathcal{W}_{nt} \setminus \mathcal{O}'$ **then**
8 $\mathcal{T}' \leftarrow$ `updateTetherConf`$(\mathsf{v}_b.\mathcal{T}_s, \mathcal{P}_{dub}(\mathsf{v}_b, \mathsf{v}_g))$
9 **if** $\phi(\mathcal{T}') \geq \phi_{des}$ **then**
10 **if** `isFeasible`$(\mathcal{T}')$ **then**
11 $\mathcal{P} =$ `bestPathAmong`$(\mathcal{V}_{op}, \mathsf{v}_b)$
12 **return** $\mathcal{P}$
13 $\mathcal{V}_{op}.remove(\mathsf{v}_b)$
14 $\mathcal{N}_c(\mathsf{v}_b) =$ `childrensOf`(v_b)
15 **for** *each* v_n *in* $\mathcal{N}_c(\mathsf{v}_b)$ **do**
16 **if** v_n *in* $\mathcal{V}_{cl}$ **then**
17 continue;
18 **if** *not* `isFeasible` $(\mathsf{v}_n.\mathcal{T}_s)$ **then**
19 $\mathcal{V}_{cl}.append(\mathsf{v}_n)$
20 continue;
21 **if** v_n *not in* $\mathcal{V}_{op}$ **then**
22 $\mathcal{V}_{op}.append(\mathsf{v}_n)$
23 **return** $\mathcal{P}$

In Algorithm 1, the function `dubinsPath` on line 8 is a subroutine that returns the shortest Dubins path between the input nodes. The function `isFeasible` returns whether the input tether configuration is feasible or not by checking any possible intersections of a tether segment with any other tether segments in the tether configuration. The function `bestPathAmong` returns a valid path, if it exists, going through any of the nodes in the open node set $\mathcal{V}_{op}$ that is better than the already found valid path going through the best node v_b, otherwise the valid path through v_b is returned. The function `childrensOf`(v) returns the set of nodes, among the neighboring nodes $\mathcal{N}(\mathsf{v}.\mathbf{v}_t)$ of the underlying tangent node, that can only be reached when robot continues its forward motion. All other nodes that require the robot to move in the reverse direction to reach these nodes are excluded by the `childrensOf`(v) function. Recall that the neighboring nodes of an underlying tangent node do not lie on the same obstacle vertex as the underlying tangent node.

Algorithm 2: Best Path among the nodes in a set $\mathcal{V}_{op}$

Input: $\mathcal{V}_{op}, \mathsf{v}_b, \mathcal{G}_t,\ f,\ g,\ h$
Output: $\mathcal{P}$

```
Function bestPathAmong (V_op, v_b):
    for v_ι in V_op do
        v_ι.P_g = dubinsPath(v_ι, v_g)
        if v_ι.P_g ⊂ W_nt \ O' then
            v_ι.T_g = T_{v_ι}(v_ι.P_g);
            f[v_ι] = g[v_ι] + h[v_ι];
            if f[v_ι] < f[v_b] then
                if φ(T') ≥ φ_des then
                    if isFeasible(T') then
                        v_b = v_ι;
    return P = v_b.P_s ∪ v_b.P_g
```

This is done to speed up the search process. Any collision-free and curvature-constrained path can only pass around the vertices of obstacles. Once we have selected a tangent node on a vertex of an obstacle, any other tangent node on that vertex will lie on the Dubins path connecting the selected node to the goal state and hence expanding to the other tangent nodes on the same vertex leads to unnecessary node expansions. That is why only tangent nodes on nearby vertices are considered for expansion.

Next, we formally discuss the condition under which completeness of Algorithm 1 can be guaranteed.

Theorem 1: For any environment with a finite number of polygonal obstacles $\mathcal{O} = \{\mathcal{O}_k \in I_o\}$, if there exists a path from the initial state $\mathbf{r}_s$ to $\mathbf{r}_g$ that satisfies the conditions 1–4 in Problem 1, then the WiTHy A* algorithm given in Algorithm 1 is guaranteed to find such a path with $\gamma_{wc} = 1$ in Eq. (7) and (9).

Proof: For $\gamma_{wc} = 1$, we have the estimate of the cost of a path going through a given node as:

$$f[\mathsf{v}_\iota] = f_{wc}(\phi((\mathsf{v}_\iota.\mathcal{T}_s)) + f_{wc}(\phi((\mathsf{v}_\iota.\mathcal{T}_g)). \tag{10}$$

This implies that the cost of the path is entirely defined based on the winding angle of the tether configuration. Since we have a finite number of obstacles in the environment, we will have a finite number of tangent nodes in the associated $\mathcal{C}^1$-tangent graph $\mathcal{G}_t$. When the same tangent node in $\mathcal{G}_t$ can be reached via different paths, the corresponding WiTHy A* graph nodes are considered to be different and hence a new WiTHy A* graph node is added to the open node list $\mathcal{V}_{op}$ (lines 22–23 in Algorithm 1). An infinite number of paths, that possibly have an infinite amount of winding around obstacles, can exist between a given node and the start node. However, under Assumption 2, the tether configurations are not allowed to have a loop in them, and hence any path from a start node to a given node that has a loop in the associated tether configuration will never have a corresponding WiTHy A* graph node added to the open list $\mathcal{V}_{op}$. This implies that there will only be a finite number of WiTHy A* graph nodes added to the open list $\mathcal{V}_{op}$. Now, any path created on $\mathcal{G}_t$ will satisfy conditions 1 and 2 if $\rho_{\bar{o}} > \max(\rho_{safe}, \underline{\rho}_{turn})$. Hence, by virtue of the finiteness of the open nodes, if there exists a path that also satisfies conditions 3 and 4, then the Algorithm 1 is guaranteed to find it in a finite number of iterations. ■

Although, as established in Theorem 1, the WiTHy A* guarantees completeness when $\gamma_{wc} = 1$, the resulting path is not necessarily the shortest of the possible paths. Finding the shortest path that solves Problem 1 requires an appropriate value of the weight γ_{wc} of the winding angle constraint cost f_{wc}. However, choosing this value may not be trivial. How to appropriately choose this weight γ_{wc} will be studied in future research.

In the absence of the winding angle constraints (i.e. $\gamma_{wc} = 0$), WiTHy A* results in a tangent graph-based variant of hybrid A*, similar to [20], where only the tangent points of inflated obstacles are expanded during the search process resulting in a significantly smaller number of expansions during the search than when using a grid of the configuration space as nodes in the standard hybrid A* algorithm [16]. Hence, WiTHy A* can generally compute the shortest path between the start and the goal state faster than the standard hybrid A*.

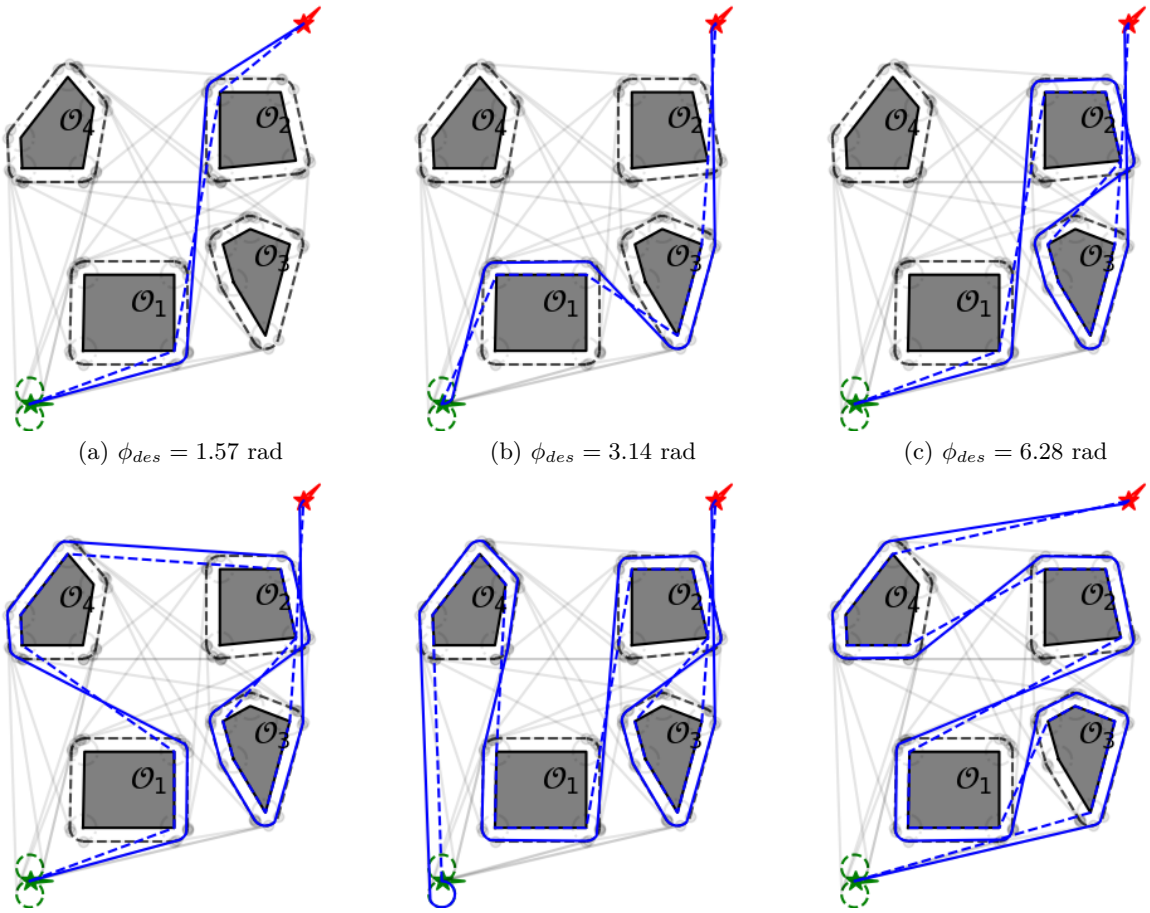
IV. Results

In this section, we study the effectiveness and performance of the proposed algorithm via simulation studies. We consider an environment with 4 obstacles defined as: $\mathcal{O}_1 = conv([6,5]^T, [12,5]^T, [12,10]^T, [6.1,10]^T)$, $\mathcal{O}_2 = conv([15,17]^T, [20,17.5]^T, [19,22]^T, [15,22]^T)$, $\mathcal{O}_3 = conv([15.9,9.5]^T, [18,6]^T, [19.6,12]^T, [17,13]^T, [15.2,12]^T)$, $\mathcal{O}_4 = conv([2,17]^T, [6,17]^T, [6.7,21]^T, [5,23]^T, [1.9,19]^T)$. The robot starts at $\mathbf{r}_s = [2.5, 1.5, 0]^T$ and the goal state is $\mathbf{r}_g = [20.5, 26.5, \frac{\pi}{4}]^T$. The paths obtained by WiTHy A* for the above parameters for various choices of desired winding angle ϕ_{des} for $\gamma_{wc} = 1$ are shown in Figure 3 and the corresponding animations (as well as additional simulations for other various scenarios) can be found at https://tinyurl.com/5n9282k5.

The actual values of the tether winding angles on these paths, the path lengths, and the numbers of while loop iterations in Algorithm 1 to find these paths are given

TABLE I: Path data for various desired winding angles for $\gamma_{wc} = 1$.

ϕ_{des}	$\phi(\mathcal{T}(\mathcal{P})))$	$l(\mathcal{P})$	Number of iterations
1.57	1.75	36.19	9
3.14	3.86	47.05	59
4.71	8.53	78.52	352
6.28	8.53	78.52	256
7.85	8.53	78.52	20
9.42	10.80	97.56	604
10.99	13.74	125.14	1330
12.57	13.74	125.14	1302
14.14	14.35	120.51	1950
15.71	no valid path	no valid path	2716



(a) $\phi_{des} = 1.57$ rad (b) $\phi_{des} = 3.14$ rad (c) $\phi_{des} = 6.28$ rad

(d) $\phi_{des} = 9.42$ rad (e) $\phi_{des} = 12.57$ rad (f) $\phi_{des} = 14.14$ rad

Fig. 3: Paths obtained by WiTHy A* for different desired winding angles (solid: path, dotted: tether).

TABLE II: Path data for various values of γ_{wc} for $\phi_{des} = 6.28$.

γ_{wc}	$\phi(\mathcal{T}(\mathcal{P}))$	$l(\mathcal{P})$	Number of iterations
0	0.48	32.08	3
0.1	6.289	68.29	343
0.3	6.289	68.29	321
0.5	6.289	68.29	282
0.7	6.289	68.29	187
0.8	6.289	68.29	129
0.9	7.26	74.41	62
1	8.53	78.52	256

in Table I. The maximum winding angle possible for the given start and goal state and given obstacles is 14.35 *rad*. As we can observe from Table I, a path is found for every desired value of winding angle below 14.35 *rad*. However, Algorithm 1 reports after 2716 iterations that there is no path found for the desired winding angle equal to 15.71 *rad* because there does not exist any path that will be able to satisfy the 15.71 *rad* winding angle on the accompanied tether configuration. Although there are some exceptions, the general trend is that as we increase the desired winding angle value, the more the number of iterations will be required to find the path.

Next, we provide the path lengths and winding angles of the paths obtained for various values of the friction cost weightage γ_{wc} for a desired winding angle $\phi_{des} = 6.28$ in Table II. As we can observe, the path obtained with $\gamma_{wc} = 1$ does satisfy the desired winding angle constraints, as expected courtesy of Theorem 1, but it is not necessarily the shortest possible path. Choosing γ_{wc} between 0.1 and 0.8 results in the shortest path that achieves the desired winding angle $\phi_{des} = 6.28$.

V. Conclusions

In this paper, we proposed a winding-constrained tangent-based hybrid A* algorithm that uses a variant of the tangent graph as the underlying graph in the hybrid A* algorithm to find the shortest path for a tethered unicycle robot with winding angle constraints on the tether configuration. The performance of the proposed algorithm is studied in simulation. It is observed that the proposed algorithm always finds a path that satisfies the user-defined winding angle constraints if such a path exists for the given start and goal states in the given obstacle environment. Furthermore, the algorithm returns a shortest path that satisfies the user-defined winding angle constraints for an appropriate value of the weightage γ_{wc} given to the winding angle constraint cost.

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] P. G. Xavier, "Shortest path planning for a tethered robot or an anchored cable," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 2. IEEE, 1999, pp. 1011–1017.

[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*, 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] 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.

[17] 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.

[18] 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.

[19] 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.

[20] M. M. Zafar, M. L. Anjum, and W. Hussain, "LTA*: Local tangent based A* for optimal path planning," *Autonomous Robots*, vol. 45, pp. 209–227, 2021.