Stability of Tethered Ground Robots on Extreme Terrains

Rahul Kumar, Vishnu S. Chipade, Sze Zheng Yong

***Abstract*— In the absence of a tether attachment mechanism that can provide infinitely large tension to the tethered robots moving on extreme planetary terrains, there is a limit on how much tension can be realistically generated or supported by the tether. In this paper, we consider a team of two robots tethered together moving on extreme terrains. The traction on the wheels of the robot and the friction between the tether and the tether attachment surfaces/objects (e.g., rocks) is the only way to support the tether tension. Given a path for the robots to navigate, we provide a systematic algorithm to check if the robots will be stable along the given path while considering the maximum constraints on the tension generated or supported by the tether. The results are validated via simulation experiments.**

I. Introduction

Planetary scientific exploration missions on extreme terrains have evolved from using traditional rovers such as Martian rovers [1], to tethered robots such as Axel and DuAxel rovers [2], TRex robot [3], to non-traditional robots such as SphereX [4]. Tethered robots being more effective on extreme terrains (steep slopes) relies on the generation of the desired amount of tension on the tether for their successful operation. Various methods listed in [5] can be used to provide tether attachment to generate the desired amount of tension to support the weight of the robot moving on extreme terrains. While many of the tether attachments mentioned in [5] are very complex to build, some other relatively less complex mechanisms involve winding the tether around certain objects to provide a certain amount of tension. Furthermore, this kind of tether attachment which involves winding the tether around objects can only support a certain amount of tension by virtue of capstan effect [6] before the tether could slide on the attachment surfaces. Significant amount of research work [7]–[11] has been done to solve motion planning problem for tethered robots on extreme terrains, however, the primary assumption in these works has been that the tether attachment can provide any amount of the tension, which may be not possible if winding around objects is the only tether attachment mechanism available.

To solve the path planning problem for tethered robots under maximum tension constraints requires careful analysis of stability to ensure that the robots can move on the designed path without getting destabilized or rolling down the inclined surfaces. In this paper, we study the problem of analyzing the static stability of a team of two robots tethered together that is moving on extreme terrains under maximum constraints on the tension that can be generated or supported by the tether wound around objects.

The authors in [7], [8] provided path planning algorithms for a tethered robot using boundary triangulated manifolds. However, the tether is assumed to be attached to a large fixed body and hence capable of supporting a large amount of tension on the tether. In [5], the authors proposed a collaborative team of an unmanned aerial vehicle (UAV) and an unmanned ground vehicle (UGV) tethered together, where the UAV flies over to the top of a cliff and winds the attached tether around a heavy object and then put a hook so that the UGV could climb the cliff by using the tension provided by the tether. The authors in [12] developed a team of parent and child rovers tethered together. The child rovers descent down a hill to examine the caves at the bottom of the hill. While both of these are effective solutions, the effectiveness heavily relies on creation of large tensions by virtue of tether attachment and not much analysis is provided on the evolution of the tether configuration and the stability of the UGV.

In this paper, we consider a team of two robots, Robot 1 and Robot 2, that are tethered together and are navigating a hilly terrain with multiple steep slopes. Robot 1 moves in the top region with a relatively flat surface while Robot 2 descends down the inclined surfaces. In the absence of an explicit attachment of the tether to a large rigid and static body, Robot 2 uses the traction on Robot 1's wheels and the friction from the objects around which the tether is wound to support its weight as it moves down the hill. Specifically, given a path consisting of waypoints to be traversed by the robots, we address the question of identifying where the tether interacts with the entities in the environment and whether the robots will be statically stable as they move along the given waypoints. Compared to other similar works, our contribution is the development of a constrained linear program to answer whether the robots will be statically stable while traversing given paths on extreme terrains subject to the maximum tension constraints induced by virtue of the traction between the wheels and the surface and the capstan effect from the tether-surface interaction. Additionally, we also provide an anchoring and de-anchoring algorithm to update the tether configuration as the robots move on the steep terrains.

R. Kumar and S.Z. Yong are with the Department of Mechanical and Industrial Engineering, Northeastern University, Boston, MA, USA (Email: `{kumar.rahul4,s.yong}@northeastern.edu`), while V. Chipade is an independent researcher (`vishnuc@umich.edu`). This work was supported in part by an Early Career Faculty grant 80NSSC21K0071 from NASA's Space Technology Research Grants Program.

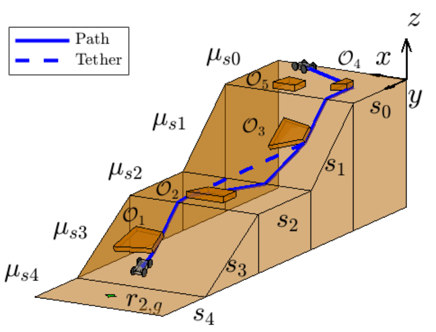


Fig. 1: Illustration of a multi-sloped terrain as well as a robot path and the corresponding tether configuration.

II. Modeling and Problem Statement

Notation: $\|\cdot\|$ denotes the Euclidean norm, while $|\cdot|$ denotes the absolute value of a scalar argument or the cardinality if the argument is a set. Further, $conv(\cdot)$ is the convex hull operator. Let $\mathbf{g}$ be the gravity vector and $g = \|\mathbf{g}\|$ denote its magnitude.

The real world has complex terrains, but for the purpose of simplifying the mathematical analysis, we consider an abstraction of the real world terrain that approximately resembles the real terrain locally. Specifically, inspired by [7], [8], we consider a structured 3-D environment $\mathcal{W}$, specifically a 2-D manifold embedded in $\mathbb{R}^3$ Euclidean space. The environment $\mathcal{W}$ consists of flat and inclined rectangular ground surfaces $\mathcal{S}_\ell = conv\left(\{\mathbf{r}^1_{s\ell}, \mathbf{r}^2_{s\ell}, ..., \mathbf{r}^4_{s\ell}\}\right)$, for all $\ell \in I_s = \{0, 1, 2, ..., N_s\}$ such that for any s_ℓ, $\ell \in I_s \setminus \{0\}$, $\mathcal{S}_\ell$ and $\mathcal{S}_{\ell-1}$ share one common edge (see Figure 1). The coefficient of friction on surface $\mathcal{S}_\ell$ is given by $\mu_{s\ell}$ and the unit vector normal to the surface $\mathcal{S}_\ell$ is given by $\mathbf{n}_{s\ell}$. These rectangular ground surfaces have obstacles $\mathcal{O}_k$, for $k \in I_o = \{1, 2, ..., N_o\}$, where the obstacle $\mathcal{O}_k$ is a tuple $\mathcal{O}_k = (\mathcal{O}^b_k, s_{ok}, h_{ok}, \mu_{ok})$ and its boundary is denoted by $\partial\mathcal{O}_k$. $\mathcal{O}^b_k = conv\left(\{\mathbf{r}^{b,1}_{ok}, \mathbf{r}^{b,2}_{ok}, \ldots, \mathbf{r}^{b,N_{v,ok}}_{ok}\}\right)$ is the polygonal base, with s_{ok} denoting the index of the surface on which $\mathcal{O}^b_k$ lies, h_{ok} its finite height along the normal direction $\mathbf{n}_{s_\ell}$ to the surface and μ_{ok} the coefficient of friction on the surface of the obstacle. Here $\mathbf{r}^{b,\dagger}_{ok} = [x^{b,\dagger}_{ok}, y^{b,\dagger}_{ok}, z^{b,\dagger}_{ok}]^T$, for $\dagger \in \{1, 2, 3, \ldots, N_{bv,ok}\}$ are the vertices of the obstacle base $\mathcal{O}^b_k$, while $N_{bv,ok}$ denotes the total number of vertices on $\mathcal{O}^b_k$. The normal to the obstacle surface at vertex $\dagger$ is denoted by $\mathbf{n}_{b\dagger,ok}$.

Further, we consider a team of two all-terrain ground robots such as TRex [3], namely Robot 1 and Robot 2, connected to each other by a tether located at a height h_t from the bottom of the wheels. The robots have masses m_1 and m_2 and the positions of the robots are given by $\mathbf{p}_1 = [x_1, y_1, z_1]^T \in \mathbb{R}^3$ and $\mathbf{p}_2 = [x_2, y_2, z_2]^T \in \mathbb{R}^3$, respectively. Each of these robots have a local coordinate frame whose x-axis points in the back direction of the robot, z-axis points in the upward direction and the y-axis points towards the right of the robot.

We make the following assumptions about the tether.

***Assumption** 1:* The tether connected to the robots are retractable and extendable, and can be ensured to stay taut during the motion of the robots.

We define a path and tether configuration as follows.

***Definition** 1 (Path):* A path $\mathcal{P} \in \mathcal{W}$ is defined as a union of path segments, $\mathcal{P} = \cup_{i=1}^{N_{ps}} \overgroup{\mathbf{r}_i\mathbf{r}_{i+1}}$, where $\overgroup{\mathbf{r}_i\mathbf{r}_{i+1}} \in \mathcal{S}_\ell$, for some $\ell \in I_s$, denotes a straight line connecting the waypoints $\mathbf{r}_i \in \mathcal{S}_\ell$ and $\mathbf{r}_{i+1} \in \mathcal{S}_\ell$ for all $i \in \{1, 2, ..., N_{ps}\}$, where N_{ps} is the total number of path segments.

***Definition** 2 (Tether Configuration):* A tether configuration that is achieved by the robots after moving on their respective paths $\mathcal{P}_1$ and $\mathcal{P}_2$ is defined as $\mathcal{T}(\mathcal{P}_1, \mathcal{P}_2) := \cup_{j=1}^{N_a-1} \overline{\mathbf{a}_j\mathbf{a}_{j+1}}$ 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.

The anchor points are of two types: 1) ones on the obstacle surfaces, 2) ones at the slope change locations on the ground surfaces. Further, we assume that the obstacles are tall enough so that the tethers can anchor on them, i.e., $h_{ok} > h_t$ for all $k \in I_o$. Note that there could be obstacles with smaller heights or even holes in the ground that the robots cannot pass through. This renders some paths going through these obstacles or holes infeasible. However, designing a path planning algorithm is not the focus of this paper and hence, these complex environments will be considered in our future work.

At each anchor point $\mathbf{a}_j = [x_{aj}, y_{aj}, z_{aj}]^T$ 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:

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

***Definition** 3 (Total winding angle):* The total winding angle on the given tether configuration $\mathcal{T}$ is given as $\phi(\mathcal{T}) = \sum_{j=1}^{N_a-1} |\phi_j|$.

Without loss of generality, we assume that Robot 1 only moves on the flat surface $\mathcal{S}_0$. This requires the tension T_1 on the side of Robot 1 to satisfy the following inequality to ensure that the Robot 1 has sufficient ground friction to prevent sliding.

$$T_1 \le F_0 + \mu_{s0} m_1 g, \tag{1}$$

where F_0 is the maximum force that Robot 1 can exert to counter the tether tension.

Then, assuming that the Robot 2 is pulling the tether, the tension T_2 on the side of Robot 2 should satisfy:

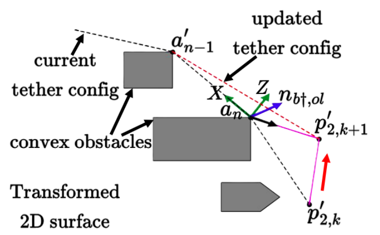
$$T_2 \le T_{2,max} = (F_0 + \mu_{s0} m_1 g) e^{\left(\sum_{j=1}^{N_a-1} \mu_{aj}|\phi_j|\right)}, \tag{2}$$

which results from tether tension amplification due to the capstan effect [6], where μ_{aj} is the coefficient of friction at the surface on which anchor $\mathbf{a}_j$ lies.

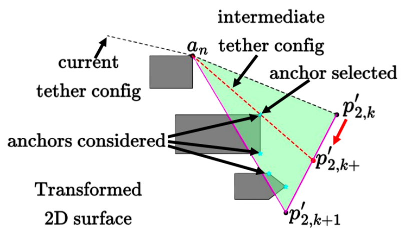
We also make the following assumption.

***Assumption** 2:* Tether-environment interaction and robot motion satisfy the following assumptions:

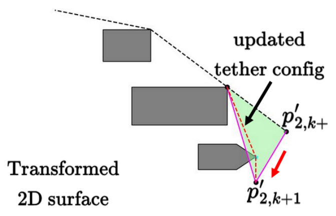
1) Obstacles' surfaces have enough surface friction to prevent the tether from sliding, once anchored.



(a) De-anchoring



(b) Anchoring



(c) Further Anchoring

Fig. 2: Illustration of tether configuration updates—Anchoring and de-anchoring

2) Tether can slide laterally on the ground surface at the slope change region.
3) The robot slowly[1] takes small steps along the given paths that can be connected by straight lines.

Next, we define the problem we consider in this paper.

***Problem** 1:* Consider the environment $\mathcal{W}$. Let $\mathbf{r}_{s1} \in \mathcal{S}_0$ and $\mathbf{r}_{s2} \in \mathcal{S}_0$ be the start positions of Robot 1 and Robot 2 respectively, and $\mathcal{T}_0$ denote the initial tether configuration. Consider paths $\mathcal{P}_1$ and $\mathcal{P}_2$ for Robot 1 and Robot 2, respectively, from their respective starting positions such that Robot 2 reaches its goal $\mathbf{r}_{g2} \in \mathcal{S}_\ell$ for some $\ell > 0$. Assume that the paths $\mathcal{P}_1, \mathcal{P}_2 \not\subset \mathcal{O}' \triangleq \{\mathcal{O}_k \bigoplus \rho_{safe} \; | k \in I_o\}$, where $\rho_{safe} > 0$ is distance from obstacle boundaries that the robots avoid for their safe motion. Design an algorithm to (i) find the intermediate tether configurations $\mathcal{T}_t$ and (ii) verify if the robots are statically stable along their paths.

III. Main Approach

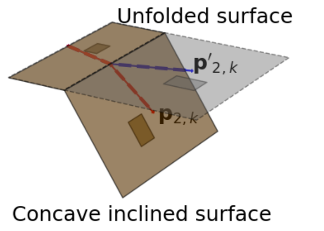
In this section, we describe the algorithms that solve Problem 1 described above.

A. Tether Configuration Update

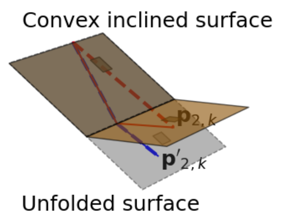
First, we address Problem 1(i) to find the tether configurations by incrementally updating the tether configurations as the robots take small steps along the given paths. Tether configuration is updated by checking for anchoring and de-anchoring events (cf. Figure 2). Anchoring predicts the attachment of the tether on possible contact surfaces to provide additional anchors and winding around the objects. On the other hand, de-anchoring predicts the detachment of the tether from the current anchors while moving from one waypoint to the next waypoint. Anchors are corners of the convex obstacles, where winding of tether is possible. Anchors also contains the interaction point with the ground at the region of slope change. Anchor prediction is critical to the stability of rover as the latest anchor prediction determines the current tether configuration which in turn control the stability of rover for next waypoint.

Algorithm 1 provides a pseudo code (similar to the one in [13]) to update the tether configuration when Robot 2 moves from current position $\mathbf{p}_{2,k}$ to next position $\mathbf{p}_{2,k+1}$ on the path $\mathcal{P}_2$. Specifically, the algorithm takes the current position ($\mathbf{p}_{2,k}$), the next position ($\mathbf{p}_{2,k+1}$), the

[1]As in [13], we consider slow motions out of an abundance of caution. Hence, we only consider (quasi-)static stability.



(a) Unfolding concave surface



(b) Unfolding convex surface

Fig. 3: Unfolding the terrain surfaces. (Solid red line: (Shortest) robot path on the terrain; Dotted red line: actual tether configuration; Solid blue line: unfolded path; Dotted blue line: unfolded tether configuration)

current tether configuration $\mathcal{T}_k$, the environment map $\mathcal{W}$ as inputs and provides an updated tether configuration $\mathcal{T}_{k+1}$ as the output. Note that the same algorithm can be used for updating the tether configuration for the motion of Robot 1 by replacing the corresponding input positions. Further, the notation $\mathcal{W}$ is overloaded to also denote a data structure storing the information about the environment (the surfaces, obstacle, etc.). Similarly, the anchor $\mathbf{a}_j$ denotes a data structure with an additional attribute named *type*, which can take the value $\mathbf{a}_j.type =$'o' if the anchor point lies on an obstacle surface or $\mathbf{a}_j.type =$'g' if the anchor point lies on the intersection of the connected ground surfaces. $\mathcal{T}_k$ can be considered as a list of anchors starting from $\mathbf{a}_0$ (Robot 1's position, with *type*='r', denoting that its a robot) to $\mathbf{a}_n$, which is the last anchor on an obstacle or ground surface.

In Algorithm 1, the function `isDetached` is used to check for tether detachment events and the function `attachAnchor` updates the tether configuration when any tether attachment event occurs. Both of these operations are performed by unfolding/rotating the required points about the surface on which the current anchor under consideration lies, which would enable us to find the shortest straight line paths for the robot. The points are then folded/rotated back to their normal position once these operations are performed. This unfolding and folding back is performed as shown in Figure 3. Note that the transformed coordinates are denoted with the superscript $'$, for example, Robot 2's position $\mathbf{p}_{2,k}$ in the unfolded space will be $\mathbf{p}'_{2,k}$.

Algorithm 1 is initialized by copying the current tether configuration $\mathcal{T}$ to $\mathcal{T}'$, which is updated in the rest of the algorithm to get the latest tether configuration. Then,

Algorithm 1: Update Tether Configuration $\mathcal{T}$

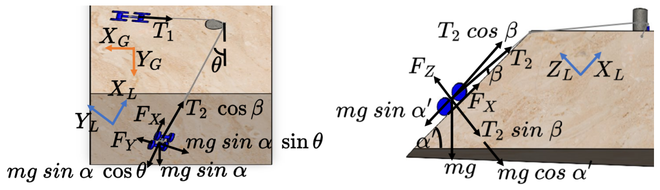
```
Input: $\mathbf{p}_k$, $\mathbf{p}_{k+1}$, $\mathcal{T}$, $\mathcal{W}$
Output: $\mathcal{T}'$
1  Function updateTether $(\mathrm{v}_s, \mathrm{v}_g, \mathcal{T}, \mathcal{W})$:
2      $\mathrm{v}_s \leftarrow \mathbf{p}_k$, $\mathrm{v}_g \leftarrow \mathbf{p}_{k+1}$, $\mathcal{T}' \leftarrow \mathcal{T}$;
3      while True do
4          $\mathrm{a}_n \leftarrow \mathcal{T}'$.pop();
5          if $\mathbf{a}_n$.type = 'g' then
6              continue;
7          if len($\mathcal{T}'$)>0 then
8              $id \leftarrow -1$;                    ▷ last item of $\mathcal{T}'$
               /* $\mathcal{T}'[id]$: $|id|$-th last item of $\mathcal{T}'$    */
9              while $\mathcal{T}'[id].type$ = 'g' do
10                 $id \leftarrow id - 1$;
11             $\mathbf{a}_{n-1} \leftarrow \mathcal{T}'$[id];
12             $\mathrm{v}'_s, \mathrm{v}'_g, \mathbf{a}'_{n-1} \leftarrow$
                 rotatewrtAnchorSurface($\mathbf{a}_n, point, \mathcal{W}$);
13             if isDetached($\mathbf{a}_n, \mathrm{v}'_s, \mathrm{v}'_g, \mathbf{a}'_{n-1}$) then
14                 $\mathrm{v}'_{s1} \leftarrow$
                     intersectLines(($\mathbf{a}_n, \mathbf{a}'_{n-1}$), ($\mathrm{v}'_s, \mathrm{v}'_g$))
15                 $\mathrm{v}_{s1} \leftarrow$
                     deRotatewrtAnchorSurface($\mathbf{a}_n, \mathrm{v}'_{s1}, \mathcal{W}$);
16                 $\mathcal{T}_{new} \leftarrow$ attachAnchor($\mathrm{v}_s, \mathrm{v}_{s1}, \mathbf{a}_n, \mathcal{W}$);
17                 if len($\mathcal{T}_{new}$) > 0 then
18                     $\mathcal{T}'$.append($\mathbf{a}_n$);
19                     $\mathcal{T}'$.extend($\mathcal{T}_{new}$);
20                 $\mathrm{v}_s \leftarrow \mathrm{v}_{s1}$;
21             else
22                 $\mathcal{T}_{new} \leftarrow$ attachAnchor($\mathrm{v}_s, \mathrm{v}_g, \mathbf{a}_n, \mathcal{W}$);
23                 $\mathcal{T}'$.append($\mathbf{a}_n$);
24                 $\mathcal{T}'$.extend($\mathcal{T}_{new}$);
25                 break;
26         else
27             $\mathcal{T}_{new} \leftarrow$ attachAnchor($\mathrm{v}_s, \mathrm{v}_g, \mathbf{a}_n, \mathcal{W}$);
28             $\mathcal{T}'$.append($\mathbf{a}_n$);
29             $\mathcal{T}'$.extend($\mathcal{T}_{new}$);
30             break;
31     return $\mathcal{T}'$
```

the last anchor $\mathbf{a}_n$ is extracted from the tether configuration $\mathcal{T}'$ (line 4). If the anchor is not on an obstacle (lines 5-6), we continue extracting the next anchors from the end of tether configuration $\mathcal{T}'$ until we find an anchor that lies on an obstacle. If, at line 7, we find that there is only one anchor present in the tether configuration, i.e.. a_0, then no detachment is possible and the algorithm only checks for attachments (discussed later in the text). If one or more anchors are available, then de-anchoring is checked by executing the `isDetached` function (line 13). Before that, in lines 8-11, we access the second last anchor in the list that is not on a ground surface by checking the anchors starting from the end of the list (line 8, where $id = -1$ denotes the end of the list and $\mathcal{T}'[id]$ denotes the $|id|$-th last item of $\mathcal{T}'$). This second last anchor is denoted as $\mathbf{a}_{n-1}$.

In the `isDetached` function, we consider the vector from $\mathbf{a}_n$ to $\mathbf{a}_{n-1}$ given as $\mathbf{X} = \mathbf{a}'_{n-1} - \mathbf{a}_n$, and the vector $\mathbf{Y} = \mathbf{n}_{b\dagger,ol} \times \mathbf{X}$, which is a cross product of the normal to the obstacle surface at the vertex † colocated with $\mathbf{a}_n$, denoted by $\mathbf{n}_{b\dagger,ol}$ and the vector $\mathbf{X}$ (see in Figure 2a). Then, the vector $\mathbf{Z}$ to the plane containing $\mathbf{X}$ and $\mathbf{Y}$ is obtained by $\mathbf{Z} = \mathbf{X} \times \mathbf{Y}$. De-anchoring takes place when the direction of vector from $\mathbf{a}_n$ to $\mathbf{p}'_{2,k+1}$ is in positive direction of the vector $\mathbf{Z}$ (see Figure 2a). If the de-attachment happens, the robot would have to travel to position v'_{s1} in the unfolded space which is obtained by the `intersectLines` function that finds the intersection point of the line from $\mathbf{a}_n$ to $\mathbf{a}'_{n+1}$ and the line from v'_s to v'_g (line 14), for which the corresponding true world position is v_{s1} obtained in line 15. During this motion, the robot may attach to other obstacles if any obstacles are present in the triangle formed by $\mathbf{a}_n$, v'_s and v'_{s1}. If such an attachment happens, the new anchor points are obtained using the `attachAnchor` function in line 16. The tether configuration is updated in lines 18 and 19, if such an attachment has happened, otherwise the current position v_s of the robot is updated to v_{s1} and the program returns to line 3. In other cases, the tether is updated based on any attachment events using the `attachAnchor` function that is discussed next.

The function `attachAnchor` predicts the updated anchor after checking for de-anchoring. As mentioned earlier, the world is transformed to a 2D plane by unfolding the $\mathbf{p}_{2,k}$, $\mathbf{p}_{2,k+1}$ and the rest of the world about the surface of the current last anchor. First, a triangle is constructed with the last anchor $\mathbf{a}_n$, current position $\mathbf{p}'_{2,k}$, and next position $\mathbf{p}'_{2,k+1}$ as the vertices (see Figure 2b), and all possible corners of the obstacles (potential anchors) that lie in this triangle are identified. The corner denoted as $\mathbf{v}_{int}$, which makes the smallest angle between the vector $(\overrightarrow{\mathbf{a}_n\mathbf{p}'_{2,k}})$ and vector $(\overrightarrow{\mathbf{a}_n\mathbf{v}_{int}})$, will be a new anchor that will be denoted by $\mathbf{a}'_{n+1}$ for subsequent processing. To attach at this newfound anchor $\mathbf{a}'_{n+1}$, Robot 2 will have traveled some distance along the path and the updated current position $\mathbf{p}'_{2,k^+}$ is determined by the function `intersectLines` which finds the intersection point of the line from $\mathbf{a}_n$ to $\mathbf{a}'_{n+1}$ and the line $\mathbf{p}'_{2,k}$ to $\mathbf{p}'_{2,k+1}$. Once $\mathbf{a}'_{n+1}$ and $\mathbf{p}'_{2,k^+}$ are found, both are de-rotated/folded to determine their actual position in the 3D world. This above process will continue with $\mathbf{a}_{n+1}$, $\mathbf{p}_{2,k^+}$ and $\mathbf{p}_{2,k+1}$ as the inputs (see Figure 2c) and will be repeated until no new anchor is detected.

Furthermore, during the folding process, when there is a surface change observed between last anchor and new anchor/goal point, then the attachment of the tether with the ground is checked at the surface change region. This is done by comparing the angle made by the surfaces with each other. If the angle is concave as shown in Figure 3a, then the intersection point of the tether with the common edge of the two surfaces in the unfolded state is computed in the real world after the folding transformation and this is added to the tether configuration as a ground anchor with *type* attribute equal to 'g'. If the angle between the surfaces is convex (Figure 3b, then no changes are made to the updated tether configuration. Note that while the anchors on ground surface can slide along the common edges of the inclined surfaces, their



(a) Free body diagram in the X_L-Y_L local frame.

(b) Free body diagram in the X_L-Z_L local frame.

Fig. 4: Schematic for stability analysis of the robots

presence plays a critical role in determining the robots' stability as these anchors change the direction of the tension generated by the tether and also contributes to tether tension amplification through the capstan effect.

The time complexity for tether configuration update algorithm is $O(nsmkd)$, where n is the number of initial anchors, s the number of surfaces in the map, m the number of obstacles, k the number of vertices per obstacle, and d the depth of recursive calls necessary to find new anchors. The depth d can be considered constant because recursive calls are made until new anchors are being detected within a triangular region, and there will be a fairly small number of obstacles present within the triangular region for sufficiently small steps.

B. Stability of the Robots

Next, we address Problem 1(ii) and describe our approach to check the static stability[1] of the robots with the updated tether configuration $\mathcal{T}$. As discussed earlier, Robot 1 is assumed to move on the flat $\mathcal{S}_0$ surface and will never slide as long as tension T_1 satisfies Eq. (1). So, we primarily focus on analyzing the stability of Robot 2. The forces acting on Robot 2 are shown in Figure 4, where the local frame (X_L-Y_L-Z_L) comprises of x-axis that is along the tether towards the anchor, z-axis that is normal to the local surface on which the rover lies, and y-axis that is perpendicular to both. The global frame (X_G-Y_G-Z_G) comprises of z-axis that is normal to the horizontal ground surface and x and y axes of the horizontal surface. In Figure 4, T_2 is the tether tension acting on Robot 2, F_X and F_Y are the ground frictional forces in the X_L and Y_L directions, respectively, while F_Z is the surface normal force acting in the Z_L direction. Note that the tether configuration $\mathcal{T}$ also determines the direction of tether force applied on Robot 2 and hence, influences its static stability.

To check if the Robot 2 is statically stable or not, we use force and moment balance equations along with the friction cones simplified as friction pyramids and tension constraints. Specifically, we formulate a linear optimization problem/program with the following set of linear constraints:

$$\sum \vec{F} = 0, \tag{3a}$$

$$\sum \vec{M} = 0, \tag{3b}$$

$$-\mu F_Z \leq F_X \leq \mu F_Z,\ -\mu F_Z \leq F_Y \leq \mu F_Z, \tag{3c}$$

$$F_Z \geq 0, \tag{3d}$$

$$0 \leq T_2 \leq T_{2,\max}. \tag{3e}$$

In the above linear program, the force balance (Eq. (3a)) is done in the local frame as can be seen in Figure 4, while the moment balance (Eq. (3b)) is done about the last anchor point. Eq. (3c) represents the constraints on the friction forces F_X and F_Y in terms of the surface normal force F_z, which has to be positive number (Eq. (3d)). Finally Eq. (3e) provides constraints on the tether tension T_2 with $T_{2,max}$ given in Eq. (2).

Robot 2 is said to be stable if there exists a feasible solution to the linear program in Eq. (3), which can be solved using a linear optimization solver such as the PuLP solver [14]. If the given path enters the unstable region with respect to the updated tether configuration, the current path is declared infeasible, and an alternate route will be checked for a feasible path.

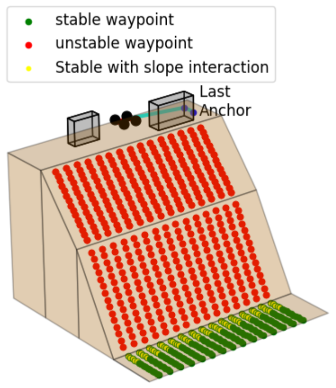
The time complexity for the stability check is $O(n+s)$, where n is the number of anchor points recorded in the anchor history and s the number of surfaces in the map.

IV. Simulations and Results

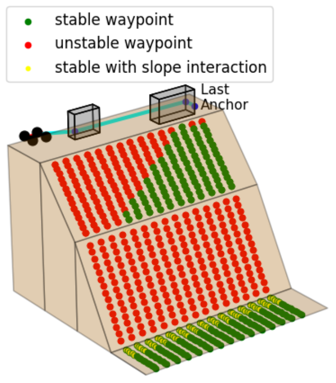
In this section, we provide simulation results showcasing the stability of the tethered robots system. For the purpose of showing the stability analysis, we consider an environment with four surfaces as shown in Figure 5 with two obstacles on $\mathcal{S}_0$ and with two successively steep surfaces $\mathcal{S}_1$ and $\mathcal{S}_2$ followed by a flat $\mathcal{S}_3$ surface. We consider the mass of both robots to be $m_1 = m_2 = 35kg$, $g = 9.81m/s2$, and coefficient of friction values to be $\mu = 0.4$. The inclined surface $\mathcal{S}_1$ and $\mathcal{S}_2$ have $55°$ and $65°$ slope respectively with respect to the top surface $\mathcal{S}_0$.

We consider three different scenarios: 1) Robot 1 partially winds around one obstacle (Figure 5a), 2) Robot 1 partially winds around 2 obstacles (Figure 5b), 3) Robot 1 partially winds around one obstacle, completes almost one round of winding around the other obstacle (Figure 5c). For each of these scenarios, we consider the initial tether configuration to have the same last anchor point from Robot 2's end. We then place the Robot 2 on various positions on the inclined surfaces $\mathcal{S}_1$ and $\mathcal{S}_2$ by sampling positions from a regular grid on these inclined surfaces. For each of these scenarios, the tether configuration from Robot 1's end is different and hence puts different constraints on the tension $T_{2,max}$. The red dots in the figure show the positions of Robot 2 at which the robot is not stable, while the green and yellow dots are stable waypoints, where the yellow ones additionally indicate that these involve tether anchoring on the slope that changes the direction of the tether force.

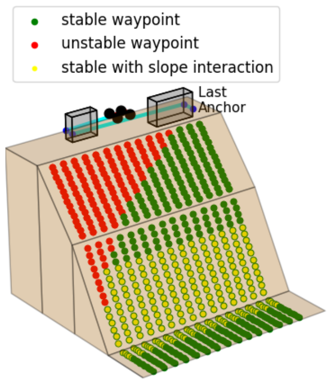
Figure 5a shows the stability results for scenario 1, in which Robot 1 only winds partially to one of the obstacles such that the maximum tension supported is $T_{2,max} = 200$. As we can see, there is not enough friction either on the wheels or due to the capstan effect to support Robot 2. In scenario 2, Robot 1 winds around more obstacles and the $T_{2,max}$ value becomes 300, for which the stability results are shown in Figure 5b. As one can see, Robot 2 is stable at the right section of



(a) Scenario 1 with tether configurations that yield ($T_{2,max} = 200$)



(b) Scenario 2 with tether configurations that yield ($T_{2,max} = 300$)



(c) Scenario 3 with tether configurations that yield ($T_{2,max} = 500$)

Fig. 5: Stability analysis of robots for various tether configurations achieved by virtue of moving Robot 1 on the surface $\mathcal{S}_0$. For Robot 2's position, red dots denote unstable position, green dots denote stable positions, and yellow dots denote stable positions when tether interaction with the slope results in additional "ground" anchors that further increases $T_{2,max}$ than depicted values due to capstan effect.

the surface $\mathcal{S}_1$ that is relatively less steep. Further, in Scenario 3, where more winding and hence a higher value of $T_{2,max} = 500$ is available, the Robot 2 is stable even on the steeper surface $\mathcal{S}_2$, as shown in Figure 5c.

Videos showing the evolution of the tether configurations for two scenarios with more obstacles for both unstable and stable paths, respectively, are available at https://tinyurl.com/mrxcc97n, which demonstrate the efficacy of our approach to solve Problem 1.

V. Conclusions and Future Work

We considered the stability of two tethered robots on extreme terrains where the only mechanism of tether attachment is winding around objects that puts a limit on how much tension can be generated and supported by the tether before the robots go into destabilizing motion. We provided an algorithm to determine the tether configurations given desired paths and a linear program to answer if the system is stable under constraints on the maximum tension values. We found that the robot navigating on the high flat surfaces will have to wind around more objects to ensure stability of the robot on inclined surfaces as the inclined surface becomes steeper. Our future work will integrate this approach with winding-constrained motion planning in [15].

References

[1] M. Ono, T. J. Fuchs, A. Steffy, M. Maimone, and J. Yen, "Risk-aware planetary rover operation: Autonomous terrain classification and path planning," in *IEEE Aerospace Conference*. IEEE, 2015, pp. 1–10.

[2] 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, and R. C. Anderson, "Axel and DuAxel rovers for the sustainable exploration of extreme terrains," *Journal of Field Robotics*, vol. 29, no. 4, pp. 663–685, 2012.

[3] 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.

[4] L. Raura, A. Warren, and J. Thangavelautham, "Spherical planetary robot for rugged terrain traversal," in *2017 IEEE aerospace conference*. IEEE, 2017, pp. 1–10.

[5] T. Miki, P. Khrapchenkov, and K. Hori, "UAV/UGV autonomous cooperation: UAV assists UGV to climb a cliff by attaching a tether," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8041–8047.

[6] F. Augugliaro, A. Mirjan, F. Gramazio, M. Kohler, and R. D'Andrea, "Building tensile structures with flying machines," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 3487–3492.

[7] 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.

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

[9] P. McGarey, M. Polzin, and T. D. Barfoot, "Falling in line: Visual route following on extreme terrain for a tethered mobile robot," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2027–2034.

[10] M. M. Tanner, *Tethered Motion Planning for a Rappelling Robot*. California Institute of Technology, 2020.

[11] L. Ren and X. Xiaotao, "Steep slope path planning for rope-tethered mobile robots in extreme terrain," in *International Conference on Information Technologies and Electrical Engineering*, 2021, pp. 1–7.

[12] J. Walker, N. Britton, K. Yoshida, T. Shimizu, L.-J. Burtz, and A. Pala, "Update on the qualification of the Hakuto micro-rover for the Google Lunar X-Prize," in *Field and Service Robotics: Results of the 10th International Conference*. Springer, 2016, pp. 313–330.

[13] 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 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 7034–7041.

[14] S. Mitchell, M. OSullivan, and I. Dunning, "PuLP: a linear programming toolkit for Python," *The University of Auckland, Auckland, New Zealand*, vol. 65, 2011.

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