An Improved Algorithm in Shortest Path Planning for a Tethered Robot

Ning Xu[*], Peter Brass[†], Ivo Vigan[‡]

1 Introduction

There is vast amount of literature focusing on motion planning for general robots. However, the same studies for tethered robots have not been investigated much. While a robot navigates in an environment with obstacles, it may meet some problems, such as a lack of power supply, or losing its wireless communication connection. If a robot is attached to a flexible tether, it can obtain sufficient power supply and stable communication through the tether.

Following [1], the shortest path planning problem for a tethered robot can be described as follows. Let E be a planar environment which consists of disjoint polygonal obstacles of n total vertices, s be the start point, and t be the destination point in the environment. Suppose a robot, modeled as a point, is attached to an anchor point u by a tether of length L. The initial configuration of the tether is considered as a polyline X of k total vertices from s to u. The goal is to find the shortest path from s to t subject to the tether length constraint.

In this paper, we assume that (1) neither the robot nor any part of the tether can enter the interior of any obstacle, and (2) the robot can cross the tether, i.e., the tether can be self intersecting.

First, we consider two different models of the shortest path planning problem. In the first model, the tether is automatically retracted and is kept taut, i.e., the tether is always the shortest path in its homotopy equivalent class. In the second model, the tether can only be retracted while the robot backtracks along the tether, because the tether may be too heavy to be dragged in some case.

[*]Graduate Center, CUNY, nxu@gc.cuny.edu
[†]City College of New York, CUNY, peter@cs.ccny.cuny.edu
[‡]Graduate Center, CUNY, ivigan@gc.cuny.edu

Figure 1 illustrates an example of the two models.

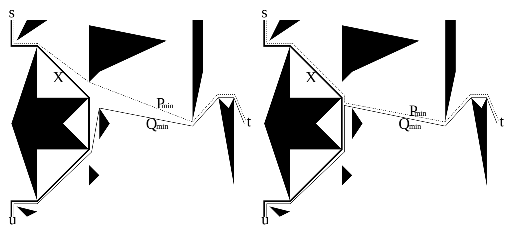

Figure 1: An example of finding the shortest path for the tethered robot. P_{min} is the shortest path, and Q_{min} is the tether configuration after the robot reaches t through P_{min}. (left) the first model; (right) the second model.

Furthermore, we consider the problem of finding the shortest path for the tethered robot to visit a sequence of target points in order. We call this problem *the sequential monitoring problem.*

Theorem 1. *If the tether is automaticaly retracted, the shortest path from the starting point s to the destination point t can be computed in $O(kn^2 \log n)$ time.*

Theorem 2. *If the tether can only be retracted while the robot backtracks along the tether, the shortest path from the starting point s to the destination point t can be computed in $O((k+n)\log(k+n)+n^2)$ time.*

Theorem 3. *The sequential monitoring problem is NP-hard when the number of target points is $O(n)$, even if the initial tether length is zero, and all obstacles are rectilinear polygons.*

The proof of Theorem 3 is omitted in the abstract.

2 The Algorithm

Let P be a path, we denote by $\overline{P}$ be the reverse path of P. If the path Q starts where the path P ends, we denote $P \circ Q$ the concatenation of P and Q.

In the first model, the tether is automatically retracted and is kept taut.

We refer to the destination point t, all bending points on X and all obstacle points as *terminals*. For a terminal v, a point c on X is an *event point* if v becomes visible at c while one moves along X from u to s. For a terminal v and an event point c, we denote by $\mathcal{P}_{c,v}$ (or $\mathcal{Q}_{c,v}$) the shortest path homotopy equivalent to the path which is the concatenation of the subpath of X from s (or u) to c and line segment (c, v), respectively. We also denote by $SP(v, t)$ the shortest path from s to t. Figure 2 illustrates an example.

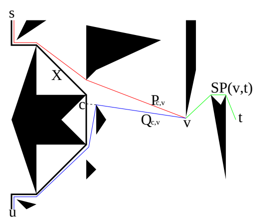

Figure 2: An example of an event point. c is an event point with respect to the terminal v. The red, blue and green path represents $\mathcal{P}_{c,v}$, $\mathcal{Q}_{c,v}$ and $SP(v, t)$ respectively.

The algorithm to find the shortest path in the first model is described in the Algorithm 1.

In the second model, the tether can only be retracted while the robot backtracks along the tether.

A point on X is *feasible* if the robot can leave from X at this point and arrives t subject to the tether length constraint.

For an obstacle point v, a point c on X is a *candidate point* if v is visible from c, and the path concatenated by the subpath of X from s to c, (c, v) and the shortest path from v to t has the length L.

The algorithm to find the shortest path in the second model is described in the Algorithm 2.

References

[1] P. G. Xavier. Shortest path planning for a tethered robot or an anchored cable. In *ICRA*, pages 1011–1017, 1999.

Algorithm 1: Algorithm SP1

```
Triangulate the environment $E$;
Compute $H(X)$ and $H(\overline{X})$ and store the funnels
  associated with each triangle on the sleeve;
Construct the Euclidean shortest path map from $t$ to
  every terminal and compute the shortest paths;
foreach terminal $v$ do
    Partition $X$ into a set of subpaths, such that each
      subpath is concave with respect to $v$;
    Compute all event points on $X$ and associate these
      points to the subpaths which contain them;
    foreach subpath do
        Use binary search to find the last event point $c$
          on the subpath with the longest $\mathcal{Q}_{c,v}$ subject to
          that $\mathcal{Q}_{c,v} \circ SP(v,t)$ is no longer than $L$;
        Compute $\mathcal{P}_{c,v} \circ SP(v,t)$, and compare its length
          with the best solution found so far. Set the best
          solution to $\mathcal{P}_{c,v} \circ SP(v,t)$ if it is shorter;
    end
end
```

Algorithm 2: Algorithm SP2

```
Construct the Euclidean shortest path map from $t$ to
  every terminal and compute the shortest paths;
If $s$ is a feasible point, directly return the path
  $X \circ SP(s,t)$ as the shortest path;
Use binary search to find the line segment of $X$ that
  contains the last feasible point;
foreach obstacle point $v$ do
    Check weather there exists a candidate point with
      respect to $v$;
    Compare with the candidate points obtained before,
      choose the point is farther to $u$ along $X$;
end
Choose the path that the robot leaves from $X$ at the best
  candidate point;
```