HOSTED BY

ELSEVIER

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

journal homepage: www.elsevier.com/locate/jestch

Full Length Article

A real-time path-planning algorithm with extremely tight maneuvering capabilities for hyper-redundant manipulators

Yalçın Bulut *, Erdinc Sahin Conkur

Department of Mechanical Engineering, Pamukkale University, 20160 Denizli, Turkey

A R T I C L E   I N F O

Article history:
Received 16 January 2020
Revised 19 June 2020
Accepted 4 July 2020
Available online 20 July 2020

Keywords:
Redundant robot
Hyper redundant manipulator
Snake robot
Obstacle avoidance
Path-planning
Confined spaces

A B S T R A C T

This paper presents a simple, effective and robust geometrical approach for path-planning of redundant/hyper-redundant manipulators in real-time within confined spaces cluttered with obstacles. A discretized path for point robots to the goal gives all the information for navigation of the manipulator. As long as it is physically possible, the employment of maneuvering space is maximized by making the links almost pinch the walls of the convoluted terrain. The maneuvering ability of the method fully utilizing maneuvering space is far beyond those of others proposed in the literature. The exemplary computer simulations verify the effectiveness of the method.

1. Introduction

There are several definitions regarding redundancy. Some of them can be stated as follows: A manipulator is called kinematically redundant when the joint space dimension is higher than that of end-effector space or if the task space dimension is less than that of end-effector space [1]. Even though they include some complexities in terms of mechanical implementation and control procedure [2], the kinematic redundancy gives a manipulator such a mobility inside the workspace so that the collisions between the manipulator and obstacles could be prevented [3] while achieving maintenance, inspection or repair tasks [4]. For the links of these manipulators not to interfere with themselves and/or obstacles around the workspace while moving to a destination, a specific path should be found [5] which the redundant or hyper-redundant manipulators, having much more joints than that of redundant robots, can follow [6]. The aim of the motion planning is to plan this specific path [7]. If a goal point for the end-effector of the manipulator to reach is given as a task, the path-planning problem arises from how to find a feasible joint path sequence. However, if the path for the end-effector to follow is already known in advance, the process of finding a feasible joint path sequence is called redundancy resolution [5]. Motion planning splits into two categories as low level and high level planning. High level planning is related to the collision avoidance while how fast this collision-free planning is carried out is the subject of low level planning [8].

Some well-known approaches that the motion planning algorithms are based on are potential field, cell decomposition and roadmap. Besides, the path-planning [9] algorithms make use of some approaches such as geometric [10] or kinematic model of the manipulator where redundancy resolution and path-planning are merged by solving a differential equation which relates the joint and task space motion [5]. Even though the classification of the path-planning algorithms in a systematic manner is somewhat difficult because of diversity in literature [11], the path-planning methods can be mainly split into four groups such as Jacobian based, curve based, geometric and path tracking approaches [12].

For obstacle avoidance, a backbone curve whose length is constant can be obtained. This curve has the ability to take the shape of the continuous hyper-redundant manipulator. The configuration of a manipulator with discrete links can be fitted to this curve as closely as possible for the manipulator to take the shape of the curve. But, the performance of the method diminishes as the redundancy of the manipulator decreases. Backbone curves having the points of large curvature values also affect the method negatively [13]. The method where numerical potential field is used navigates the discrete links of the manipulator through obstacles without any collision by taking advantage of the gradient descent.

* Corresponding author.
E-mail addresses: ybulut@pau.edu.tr (Y. Bulut), sconkur@pau.edu.tr (E.S. Conkur).

Peer review under responsibility of Karabuk University.

https://doi.org/10.1016/j.jestch.2020.07.002
2215-0986/© 2020 Karabuk University. Publishing services by Elsevier B.V.

Nomenclature

GL	Guide link	arLink[n]	Link array
RL	Reference link	TP	Tangent point
DS	Downstream	AL	Adjacent link
US	Upstream	SPC	Starting point coordinates
n	Link number	EPC	End point coordinates
rfn	Reference link number		

The algorithm makes the field create virtual torque at the joints of the manipulators. The method gets rid of the complex inverse kinematics, local minima, and it is real-time. The performance of the method is not affected by the number of the links. But it suffers from saddle-point traps and it is computationally intensive as it necessitates the number of control points on a link to be as high as possible [14]. The potential field of the workspace can be combined with the energy principle for path-planning of a snake robot. Lagrange's equations of motion provide collision-free paths. Because each link of the manipulator is modeled with the pair of spring and damper, it causes the link lengths to change. Also, the algorithm includes the problem of self intersection of the links [15]. There is also a similar paper where the rigid links of the redundant manipulator is approximated by the virtual springs [16]. Another path-planning method proposes a method that the posture of the hyper-redundant manipulator is configured with the aid of curvilinear theory, so called serpenoid curve. A search for obstacles is carried out through changing the coefficients defining the curvature function. This procedure takes a long time, but thereafter the generation of collision-free path is real-time [17]. An obstacle inside the workspace can also be avoided by utilizing a switching objective function. The motion of the manipulators in free-space (outside the confined environment) is planned by using the method of elastic model [18]. A proposal for combining backbone curve with Generalized Voronoi Graph (GVG) prescribes a follow-the-leader strategy for the path-planning. The most distal link of the hyper-redundant manipulator guides the links which come after it to where it passes.

When it comes to the manipulators whose link lengths are relatively high with respect to their widths, the mapping of these links to this curve gets harder and is prone to result in intersecting with the obstacles [19]. A path-planning method where pseudo-inverse velocity control is combined with the attractive poles concept can be used in enclosed workspaces [20]. There is also a motion planning method based on sensory data [21]. Another proposal introduces Bump-surfaces notion representing the obstacles to be avoided. This approach is utilized to plan the path of the redundant manipulator so that the whole arm doesn't collide with the obstacles in the constrained terrains [22]. A path-planning algorithm called BFA (Backtrack-free path-planning) finds the path for each link of the redundant manipulator individually in Euclidean space. The computation time of the method increases linearly with the number of links on condition that the same relationship between the path length and the link numbers should also be ensured. However, the algorithm doesn't take into account the self-intersection of the links [23,24]. Another algorithm for collision avoidance path-planning (CAA) is also carried out in Euclidean space. A Minimum Distance Technique (MDT) calculates the minimum distances between each link and between the links and obstacles. Therefore, it doesn't allow the self-intersection of the links as well while the manipulator reaching the goal without colliding with the obstacles [25].

The path of a snake robot whose each module is comprised of Stewart platform is planned using a gait fitting algorithm which maps the configuration of the manipulator to the serpenoid curves. Stewart platform is actuated digitally by pneumatic cylinders having only fully extended or retracted state at a time, namely there aren't any intermediate states to control. Since the algorithm is only effective for the robot to follow the paths of relatively small curvatures, it is enforced to add more modules for the manipulator to follow the large curvatures [26]. The path for the hyper-redundant manipulator of variable length is divided into sub-paths, each represented by the perimeter of a half ellipse, then the number of links needed for approximating this sub-paths are calculated. Joint angles of the links are acquired by inverse kinematics [27]. A technique using particle swarm optimization for path-planning (PASO) of hyper-redundant manipulators in 3D environment with randomly cluttered obstacles finds suitable waypoints using a fitness function after it searches and finds proper joint angles to the manipulator's inverse kinematics [28].

A hyper-redundant arm with discrete links, each of which is actuated by electromagnetic energy in a bi-stable manner i.e., tilted to the left or right has been offered. It utilizes the follow-the-leader approach for advancing the manipulator through the path. Its kinematic model relies on Denavit-Hartenberg parameters. A relationship for measuring the performance of different switching patterns to follow the path in planar space is established based on the least squares deviation and optimized [29], and it's extended to 3D environment [30]. The path-planning of a hyper-redundant manipulator design [31] whose each discrete module is actuated by cables has been presented. The deployment of the robot is approximated by serpentine curves. Given the curvature function of this curve, the path without any collision is obtained and the manipulator is mapped to this curve. The drawback is that relatively small joint angles limit the maneuvering capability of the robot [32]. Another paper introduces sampling-based motion planning for the manipulators to advance along the tubes [33]. The path-planning problem of a serpentine arm with discrete links is dealt with geometrically by means of the follow-the-leader concept based on the backbone curve approach [34]. A path following method employs polynomial splines to create a collision-impeded continuous path. Then, the path is divided into discrete points. An algorithm, which is coined as prediction lookup, estimates the path points on which the link joints should match. Any matching errors which have to do with the discretization of the continuous path are compensated by an interpolation algorithm subsequently [12].

Our previous work for path-planning of redundant manipulators are summarized as follows: A geometric approach was presented for path-planning. The links of the hyper-redundant manipulator and obstacles are modeled as lines and ellipses, respectively. When the control algorithm senses any intersection between the links and obstacles, it repulses the links from the ellipses. The tight maneuvering capability of the method measured with respect to an index defined in the paper is relatively poor in [35]. But, it was improved in [36] which presents an algorithm based on the beam analysis enhancing the tight maneuvering ability of the manipulator. Then, it was thought that the path could be approximated by B-spline curves. The algorithm developed keeps the links of the hyper-redundant manipulator approximately tangent to these curves, thereby follows the path successfully [11]. Since B-spline curves have some limitations that stems from having analytical equations, a discretized path was proposed, which consists of points that are close enough to each other starting from the manipulator base to the goal [37]. Using master link concept defined in [37], the manipulator achieved to follow the path smoothly.

Although path following or mapping manipulator links on the path in a variety of ways is an effective tool for path-planning, it inherently has a major limitation; the manipulator cannot follow the path properly when its link lengths get longer relative to the curves of the path. Therefore, the manipulator gets stuck among obstacles although there is plenty of space to navigate. This paper attempts to solve this problem by combining the beams in [36] and the path consisting of points in [37], which results in extremely tight maneuvering capabilities for redundant/hyper-redundant manipulators. The superiority of the method stems from using the maneuvering space effectively. The geometric approach makes it simple, robust and computationally inexpensive, thus very fast and reliable. The path-planning ability of the method proposed is much above not only that of our previous studies but also other methods presented in the literature according to a benchmark scheme which will be explained later in this paper.

The remainder of this paper is summarized as follows: Section 2 explains how the global path is obtained while the path-planning method is introduced at Section 3 in all aspects. Section 4 includes exemplary computer simulations to display the effectiveness of the method. In Section 5, tight maneuvering performance of our method is revealed based on a benchmark scheme and the performance of the method is compared with other papers presented in the literature. Section 6 reports the conclusions.

2. Obtaining the global paths and general concepts

The Laplace equation in Eq. (1) defines a potential of $\phi(r)$, where $\phi(r)$ represents the value of potential field of a scalar point, r. Eq. (1) is on a closed, continuous region of space represented by a grid. The boundaries of the obstacles and the goal point form the boundary of this space. The laplace equation under Dirichlet boundary conditions in 2-D environments having connected grids with equal spaces can be represented by the partial difference equation described in Eq. (2), where i and j means grid position in the x and y direction, respectively.

$$\nabla_2\phi = 0 \tag{1}$$

Eq. (2) enables an iteration procedure to be carried out on the grid. The iteration is performed to get field values at all points on the grid after the goal point, the boundary points and the rest are given a value of 2^{-126}, zero and one, respectively. A field value at any point in the region is obtained by means of the linear interpolation.

$$\phi_{(i,j)} = \frac{(\phi_{(i+1,j)} + \phi_{(i-1,j)} + \phi_{(i,j+1)} + \phi_{(i,j-1)})}{4} \tag{2}$$

The Eq. (3) gives the direction of the largest descent. Thus, a unique path consisting of points for a point robot to avoid obstacles while reaching the goal is obtained [36].

$$\alpha = atan2\left(\frac{\phi_{(i,j-1)} - \phi_{(i,j+1)}}{\phi_{(i-1,j)} - \phi_{(i+1,j)}}\right) \tag{3}$$

Using the principles explained so far, the path points are obtained numerically as shown in Fig. 1. Although we use numerical potential field, it is not mandatory. A similar method can be made use of to obtain the path. Once the points are obtained, they need further numerical kind of treatment, which is called windowing, to make the path smooth as much as possible. That is, each coordinate of the path points are settled down by an average of the coordinates of the consecutive path points before and after itself. As the number of consecutive path points escalates, whole path converges to a continuous curve. Fig. 1-a shows the raw path while Fig. 1-b shows the smoothed path exposed to the windowing treatment.

Some concepts which will be needed later in this paper are also shown in Figs. 2 and 3. Beams [36] are the lines drawn perpendicular to the tangent lines corresponding to every path points. When moving towards the goal, the beams are called red or green beams if they are on the right or left side of the points, respectively.

Critical corner detection is the crucial for the path-planning algorithm to make its decisions automatically during the execution of the program. The curvature, the red and green beam lengths are calculated for each path point. If the red or green beam lengths are under a pre-defined value and the curvatures are above a pre-defined magnitude, the path points conforming to these conditions constitutes critical corners as shown in Fig. 3.

There are also semi-critical corners resulting from obstacle-1 and obstacle-2 as shown in Fig. 4. Their pre-defined values for beam length and curvatures are different than those of critical corners.

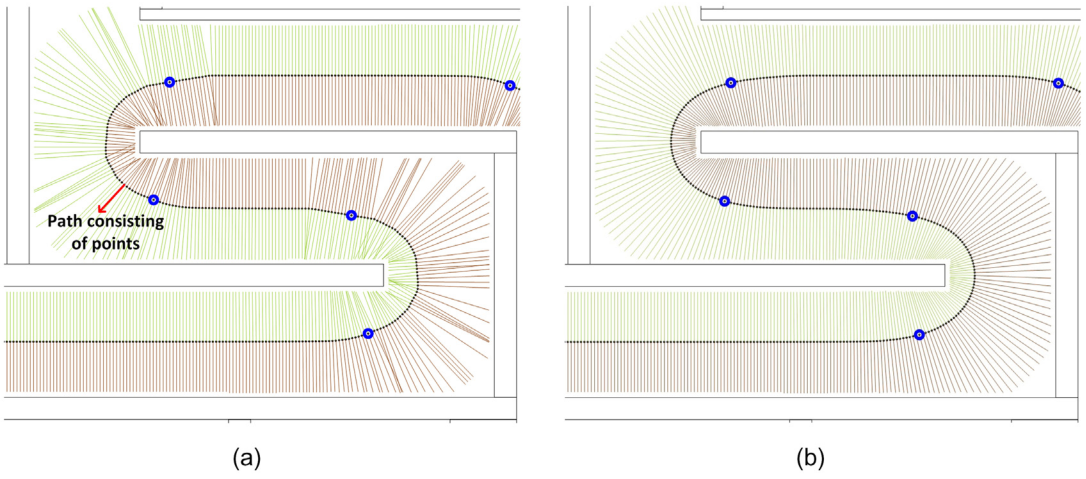


Fig. 1. Windowing of path points. a. Raw path, b. Smoothed path.

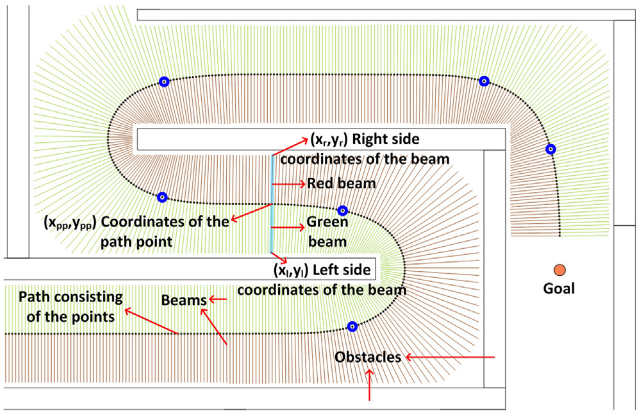

(x_r,y_r) Right side coordinates of the beam
Red beam
(x_{pp},y_{pp}) Coordinates of the path point
Green beam
(x_l,y_l) Left side coordinates of the beam
Path consisting of the points
Beams
Goal
Obstacles

Fig. 2. Concepts of the path-planning algorithm.

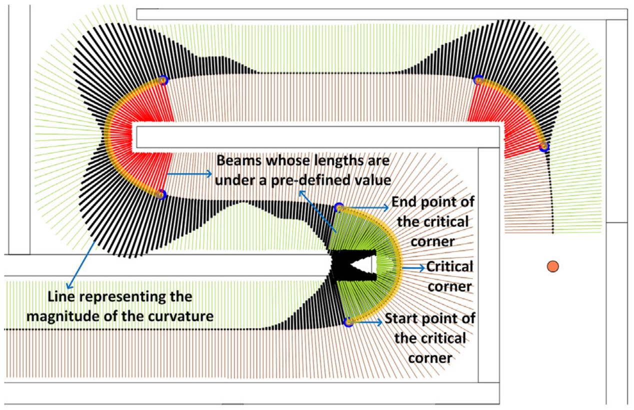

Beams whose lengths are under a pre-defined value
End point of the critical corner
Critical corner
Line representing the magnitude of the curvature
Start point of the critical corner

Fig. 3. Detection of the critical corners.

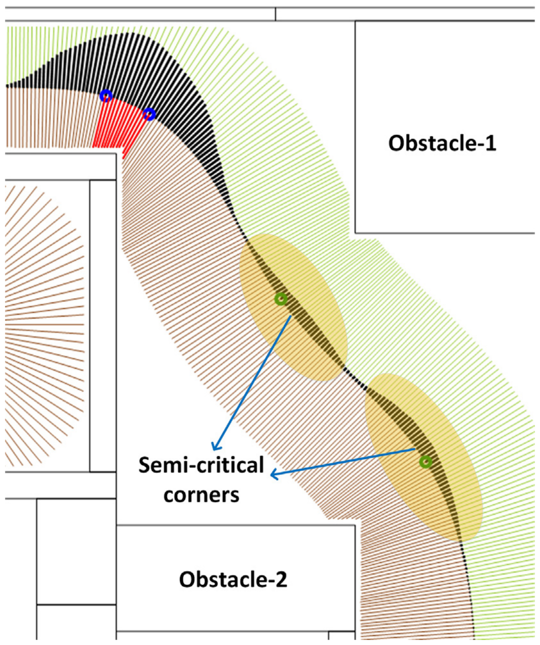


Fig. 4. Detection of the semi-critical corners.

3. Path-planning method

In this section, path-planning method will be explained in detail.

3.1. Finding a tangent point along the path

A vector, $\overrightarrow{F}_1$, is drawn from the starting point of the related link to the path point closest to the starting point of that link. Another vector, $\overrightarrow{F}_{2}$, is also drawn in the same way using the path point next to the previous one, as shown in Fig. 5a, and vector multiplication of those two vectors is calculated. The same calculation is repeated increasing the indices of path points by one. Once the result of vector multiplication changes sign, the path point after which the sign change occurs is referred to as tangent point as can be seen from Fig. 5b.

3.2. Finding the path point to which the end point of the link is closest

The portion of the link between tangent point and end point is used to search for the closest path point to the end of the link. Each path point is assumed to have its own tangent line. The beams are the ones that is perpendicular to these tangent lines. The link and beams have some intersection points as shown in Fig. 6. Starting from the intersection point which is closest to the tangent point, the distances between intersection points and end point are calculated. When the distance is under a pre-defined threshold value, the calculation is terminated. The path point through which the beam causing the calculation to be terminated passes is considered to be the closest path point to the end of the link.

3.3. Finding the closest edge side coordinate in case of no tangency

If there isn't any tangent point left for the related link, the algorithm finds the closest edge side coordinate meeting the link length constraint as shown in Fig. 7.

Beginning from the path point to which the starting point of the related link is closest**,** a vector is drawn from the starting point of the link to the leftmost side of the beam corresponding to that path point as shown in Fig. 8. The magnitude of this vector is calculated. If this magnitude is not greater or equal to the link length, the indice of the path point is increased by one and the process is repeated until the condition has been met. Once the condition is met, a unit vector is obtained between the starting point of the link and the coordinates of the leftmost side of the beam enabling the condition to be met. The end point coordinates of the link is calculated adding the result of the unit vector multiplied by the link length on the starting point coordinates of the link.

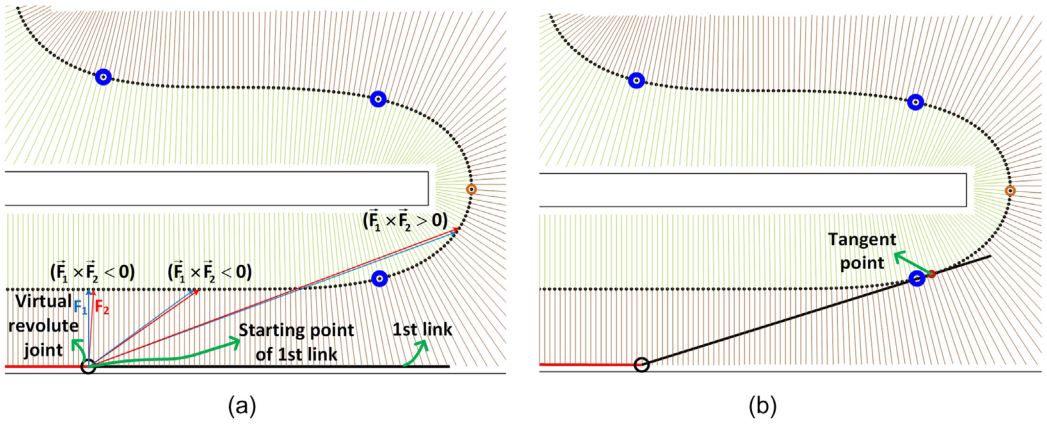
3.4. Finding the closest path point coordinate in case of no tangency

Near the semi-critical corner region, if there isn't any tangent point for the link and the algorithm finds the closest edge side coordinate meeting the link length constraint, the link interferes with the obstacle as shown in Fig. 9.

The solution to this problem is to make the algorithm find the closest path point coordinate meeting the link length constraint instead of the closest edge side until the link gets away from the semi-critical corner as shown in Fig. 10.

3.5. Corner identification

Two vectors whose initial and terminal points are path points inside the corners are created. If the vector multiplication of these two vectors are positive or negative, the corner is called right cor-



$(\vec{F}_1 \times \vec{F}_2 > 0)$
$(\vec{F}_1 \times \vec{F}_2 < 0)$
$(\vec{F}_1 \times \vec{F}_2 < 0)$
Virtual revolute joint
F_1
F_2
Starting point of 1st link
1st link
Tangent point
(a)
(b)

Fig. 5. a-b. The method of finding a tangent point along the path.

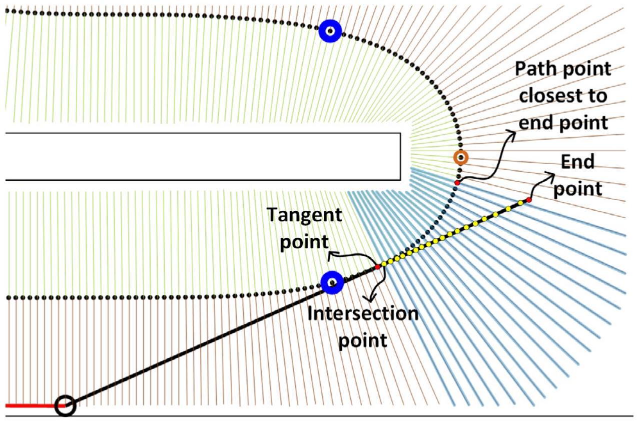


Fig. 6. The method of finding the path point to which the end point of the link is closest.

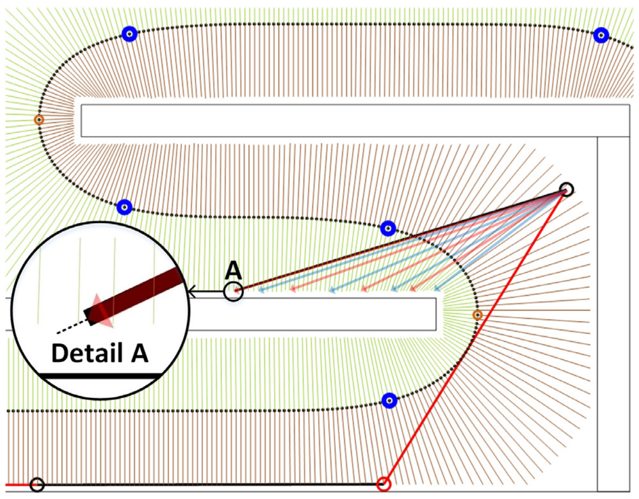


Fig. 8. The detail of finding the closest edge side coordinate in case of no tangency.

ner or left corner, respectively, as shown in Fig. 11. After there isn't any tangent point for the link to find, the algorithm determines the closest edge side coordinate with respect to the identification of the following critical corner. If this critical corner is identified as right or left corner, the link takes its position at the right or left side of the beam, respectively.

3.6. Blocking the ability of a link to find tangent to the path

When the end point of the second link (Fig. 12a) has jumped above a threshold distance from A to A′ (Fig. 12b) inside the critical corner space due to the geometrical constraints, the ability of the third link to find a tangent point should be blocked (Fig. 12c) until second link finds a tangent point. Otherwise, the second link would begin to interfere with the obstacle. Therefore the end point of the third link jumps from the point of B′ to B″. The point B″ is the closest edge side coordinate in case of no tangency. During the jumping from the point A to A′, the tip of the link does not collide with the obstacle. This can be seen from the Fig. 12d where a circle with a radius of link length and center of the point B′ proves that there is not collision. Besides, the ability of the second link to find a tangent point is also blocked if the first link is already tangent to a path point.

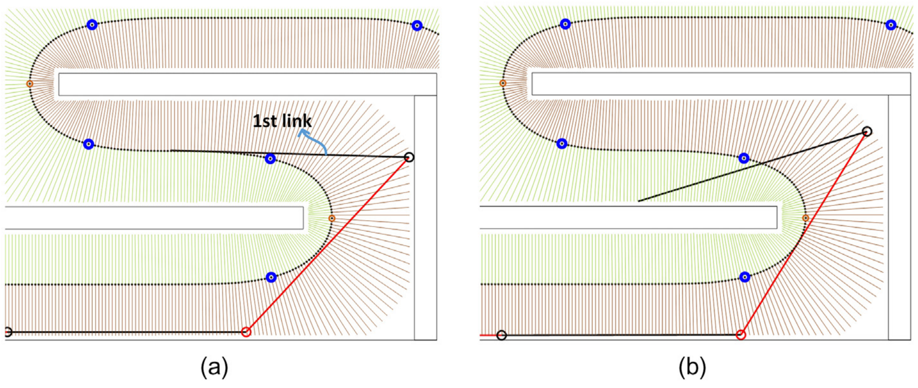


Fig. 7. a-b. The method of finding the closest edge side coordinate in case of no tangency.

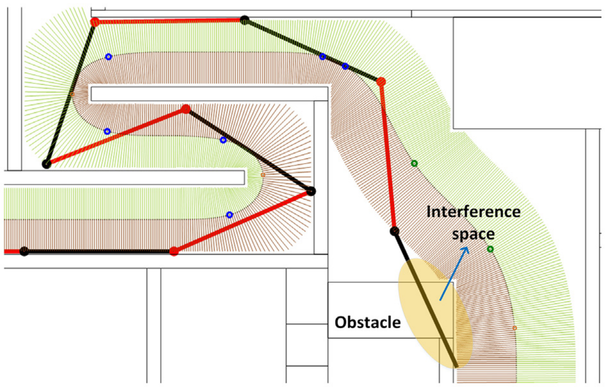


Fig. 9. The interference space.

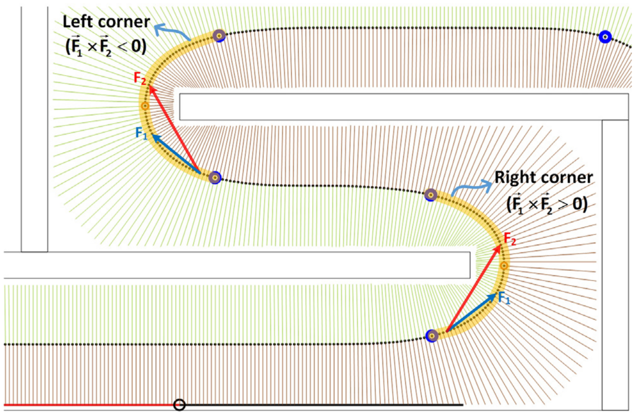

Left corner ($\vec{F}_1 \times \vec{F}_2 < 0$)
F_2
F_1
Right corner ($\vec{F}_1 \times \vec{F}_2 > 0$)
F_2
F_1

Fig. 11. Right or left corner identification.

3.7. Softening the sudden jumps of the links

Sudden jumps from tangent points to edge side points and vice versa should be softened to ensure the continuity of the motion of the links. The end point of the first link would have jumped from the point A to B without the softening algorithm as shown in Fig. 13. For softening the motion, first of all, a vector of $\overrightarrow{F}_1$ whose initial and terminal point are A and B, respectively, is created. After the unit vector of $\overrightarrow{F}_1$ is obtained, the distance between A and B is divided into as much equal pieces as the softening degree. The value of one piece is multiplied by the unit vector of $\overrightarrow{F}_1$ to get a vector of $\overrightarrow{F}_2$. The first link is tangent to the terminal point of $\overrightarrow{F}_2$.

The same principle is applied to the first link as it moves until the end point of it reaches the leftmost side of the beams as shown in Fig. 14.

3.8. Flowchart of the path-planning algorithm

Refer to nomenclature section of this paper for abbreviations used in Figs. 15 and 16. The Fig. 15 shows the snapshot of the path-planning software for six links with abbreviations used on the flowchart depicted in Fig. 16 for the clarification of the algorithm. We have split the flowchart into three different parts, namely, Part A, B and C which represent the sections of the algorithm for the reference link, downstream links and upstream links, respectively. It is assumed that the manipulator has enough number of links to reach the goal. The collision-free configuration of each link is calculated individually. The dimensions of the virtual revolute joints haven't been taken into account.

4. Simulations

Our method, which has been tested on four different workspaces on computer, gives the manipulator with discrete links high maneuvering capability as shown in Fig. 17. Fig. 17a depicts the most challenging environment in which all maneuvering spaces are extremely tight regarding the lengths of the links whereas Fig. 17b, c and d include also light maneuvering spaces, indicating the method is applicable to path-planning of a general case. The beams in Fig. 17a are intentionally shown for the whole arm planning explained in Fig. 16 to be confirmed.

The plot depicted in Fig. 18a establishes a near-linear relationship between path length and time based on the environment shown in Fig. 17a. The total number of whole-arm configurations to reach the goal is 1870 and the total length of the path is 10,505 pixels. The manipulator reaches the goal in 1921 ms, which means the average duration for each pose is 1.03 ms, and thereby the method is real-time. The time taken for each new configuration, which is just in milliseconds, is shown in Fig. 18b.

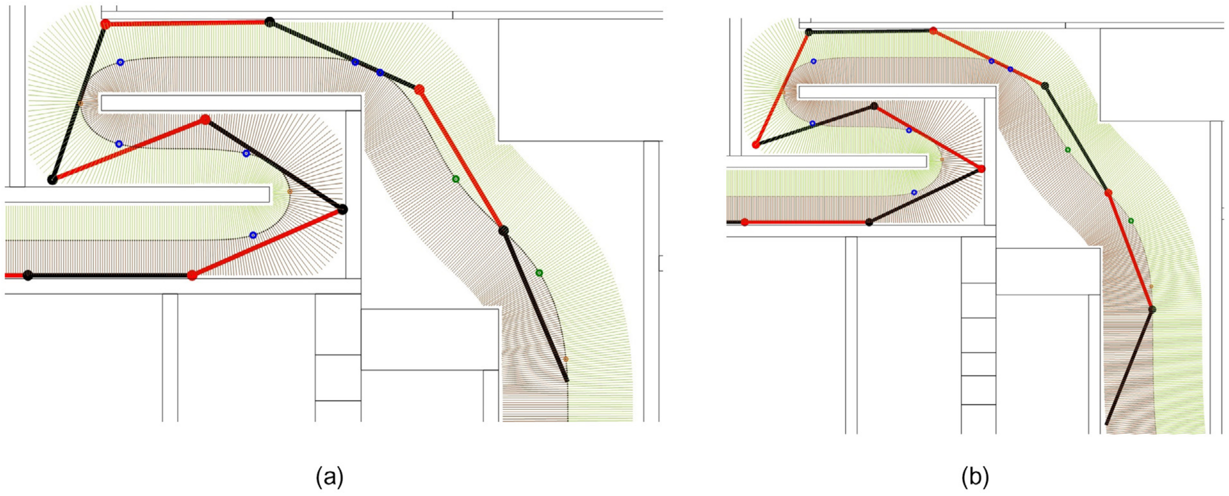


Fig. 10. a-b. The solution to interference.

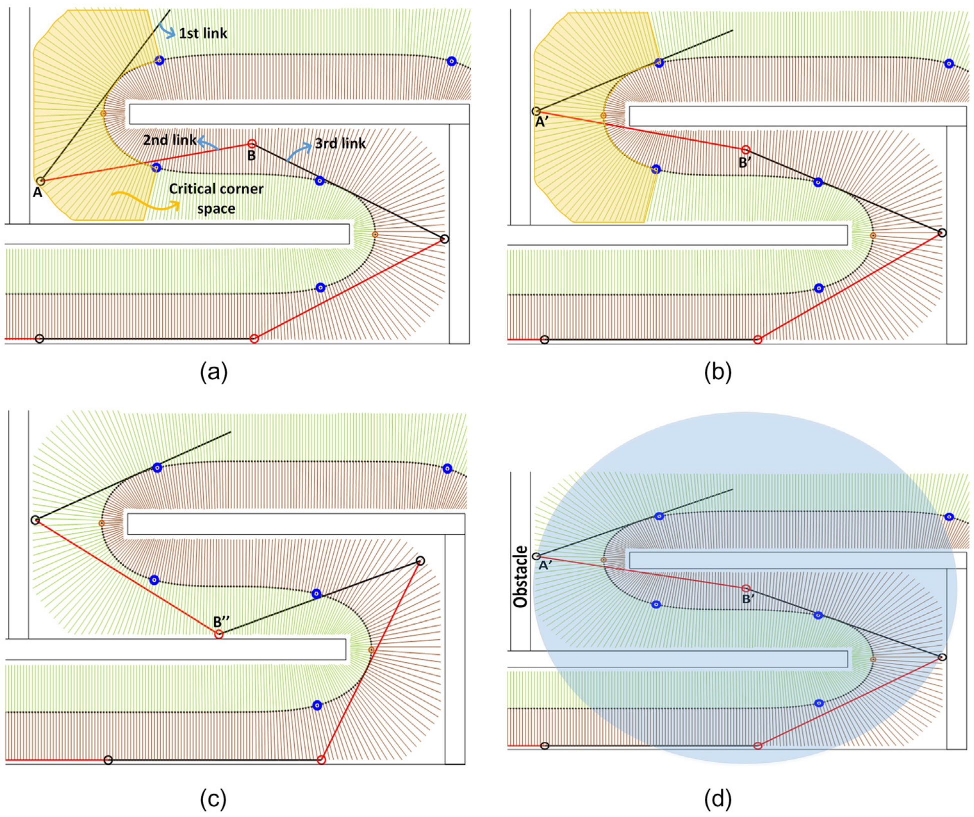


Fig. 12. a-d. Blocking the ability of a link to find tangent to the path.

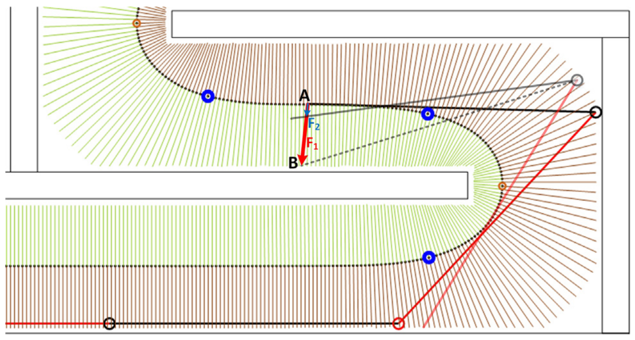


Fig. 13. The method for softening the sudden jumps.

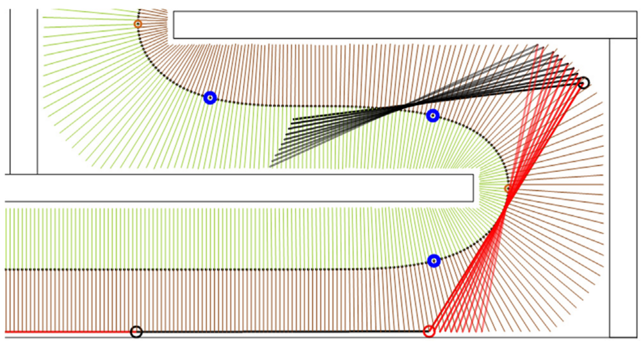


Fig. 14. The softening process.

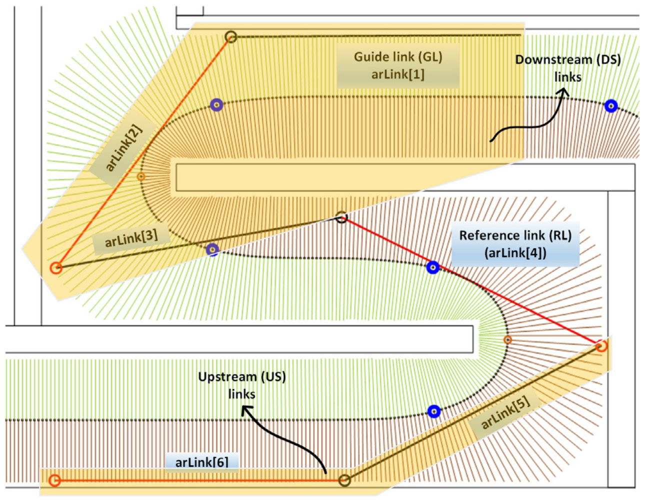


Fig. 15. Concepts for the clarification of the algorithm.

The software has been developed in C Sharp and performed on Intel® Core™ i5-4210U CPU 2.40 GHz laptop. A simulation video is shown in Video 1. Time delays between each configuration have intentionally been added for the sake of the clarification of the algorithm.

5. Discussion

The distinguishing features of the method have been discussed below.

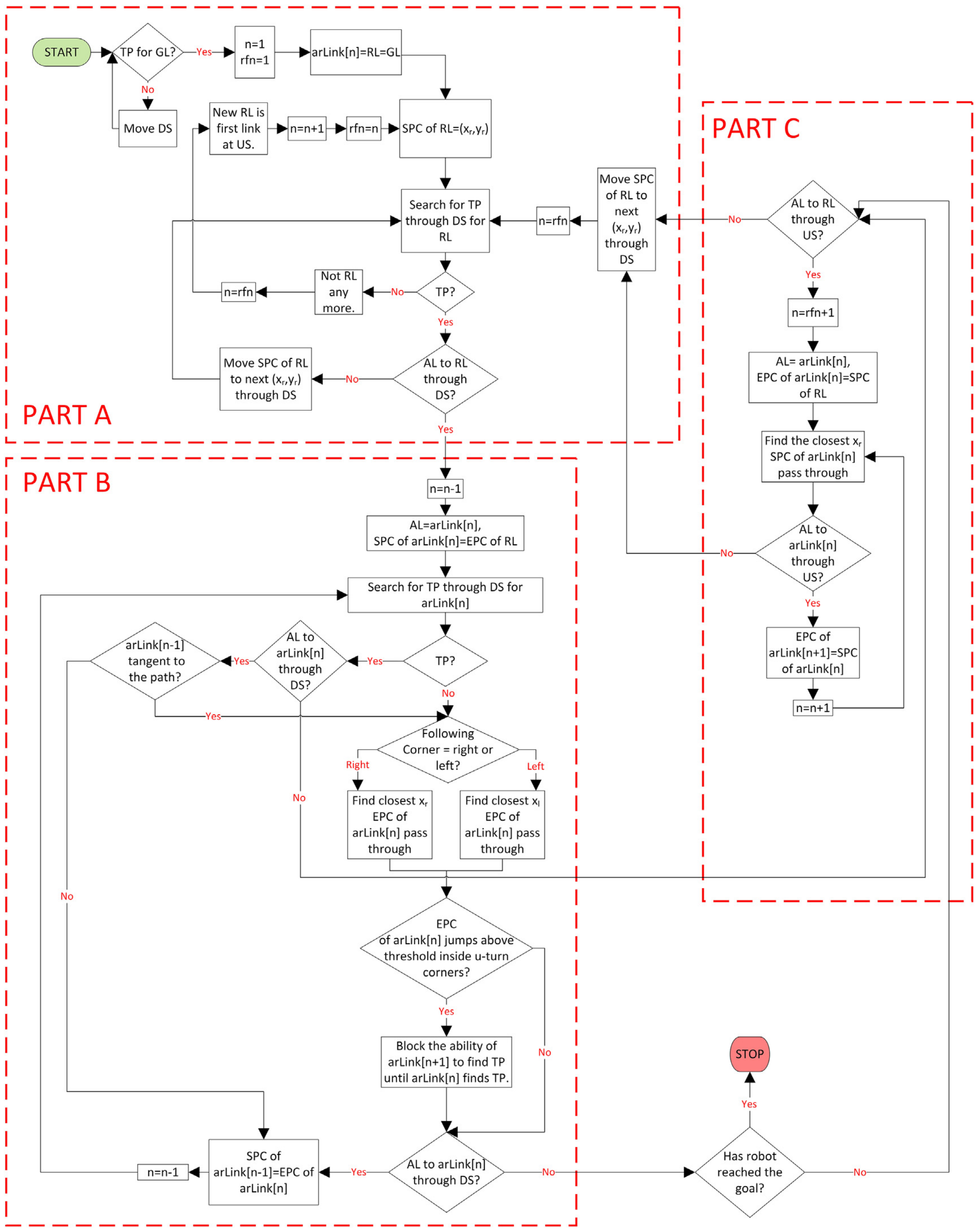

START
TP for GL?
Yes
No
n=1
rfn=1
arLink[n]=RL=GL
Move DS
New RL is first link at US.
n=n+1
rfn=n
SPC of RL=(xr,yr)
Search for TP through DS for RL
n=rfn
Move SPC of RL to next (xr,yr) through DS
TP?
Not RL any more.
n=rfn
AL to RL through DS?
Move SPC of RL to next (xr,yr) through DS
PART A
PART B
PART C
AL to RL through US?
n=rfn+1
AL= arLink[n], EPC of arLink[n]=SPC of RL
Find the closest xr SPC of arLink[n] pass through
AL to arLink[n] through US?
EPC of arLink[n+1]=SPC of arLink[n]
n=n+1
n=n-1
AL=arLink[n], SPC of arLink[n]=EPC of RL
Search for TP through DS for arLink[n]
TP?
AL to arLink[n] through DS?
arLink[n-1] tangent to the path?
Following Corner = right or left?
Right
Left
Find closest xr EPC of arLink[n] pass through
Find closest xl EPC of arLink[n] pass through
EPC of arLink[n] jumps above threshold inside u-turn corners?
Block the ability of arLink[n+1] to find TP until arLink[n] finds TP.
SPC of arLink[n-1]=EPC of arLink[n]
n=n-1
AL to arLink[n] through DS?
Has robot reached the goal?
STOP

Fig. 16. The flowchart of the path-planning algorithm.

5.1. Maneuvering ability

Since the hyper-redundant arms have high degrees of freedom, the management of their configuration while they advance through confined spaces has become key issue for decades. We have come up with a novel method to deal with this problem. It has high maneuvering ability. As long as it is psychically possible, it maneuvers successfully with minimum number of links. Therefore, it utilizes free space fully. To express the maneuvering ability mathematically, a benchmark environment was introduced in [36], which is also depicted in Fig. 19. For this purpose, an index I has been presented as shown in Eq. (4), where $l_{\max}$ and l_{link} represent the maximum link length that would be geometrically possible and each link length, respectively. In our new method, all discrete links are the same length of 430 pixels and $l_{\max}$ is equal to 440 pixels, which results in an index value of 0.98. It is inferred that our method does not waste almost any maneuvering space provided that it is physically possible to maneuver.

$$I = l_{link}/l_{\max} \tag{4}$$

The tight maneuvering space dimensions in pixels is shown in Fig. 19. The lengths of l_1, l_2, l_{link}, and $l_{\max}$ in pixels are 200, 200, 430, and 440, respectively. The length of l_1 and l_2 should be no shorter than these values.

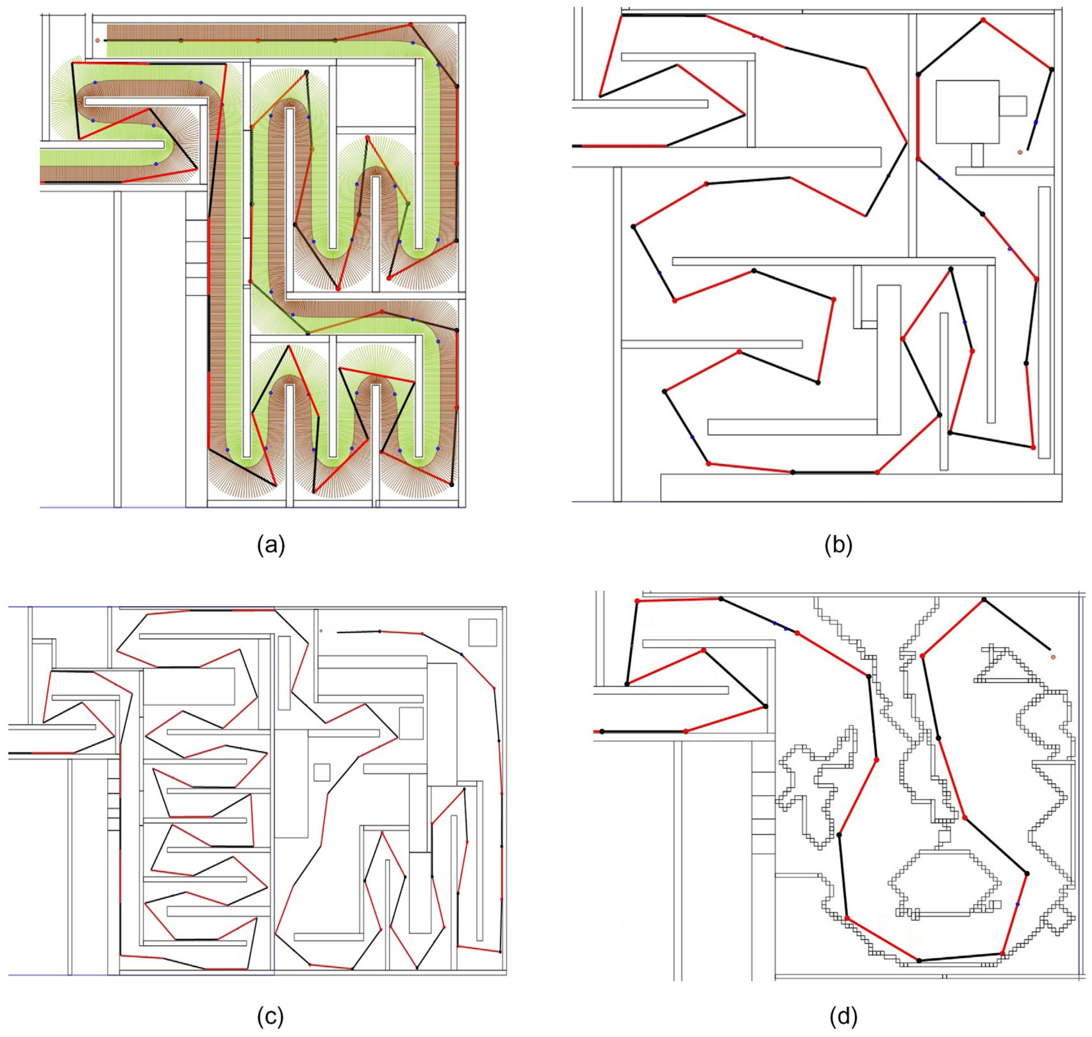


Fig. 17. a-d. The execution of the path-planning algorithm.

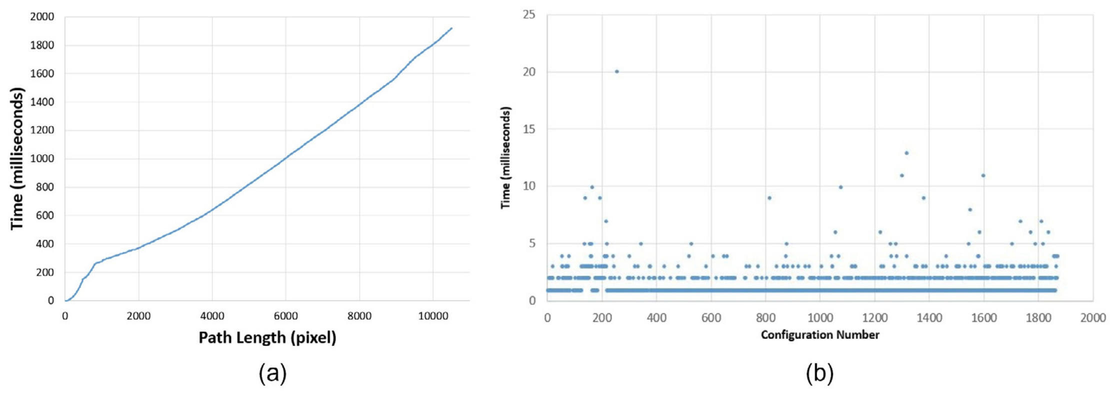


Fig. 18. a-b. Simulation results.

5.2. Number of DOFs

Although it maneuvers with minimum number of links, the performance of the method is not limited to certain number of links. On the contrary, its complexity drives up almost linearly (see Fig. 18a) with the number of links, and its performance does not depreciate in any way. The number of the links is almost limited with the power of the computer.

5.3. Complexity and extensibility

It is simple and robust method. Therefore, it is very fast and reliable. The methods such as the one in [35] waste time since links are repeatedly repelled by obstacles. Erratic motion is not a rare case in such arrangements as well as employing large number of links is hard. In our method, after having the path and the beams, a link's motion is simple to choose one out of a few options and

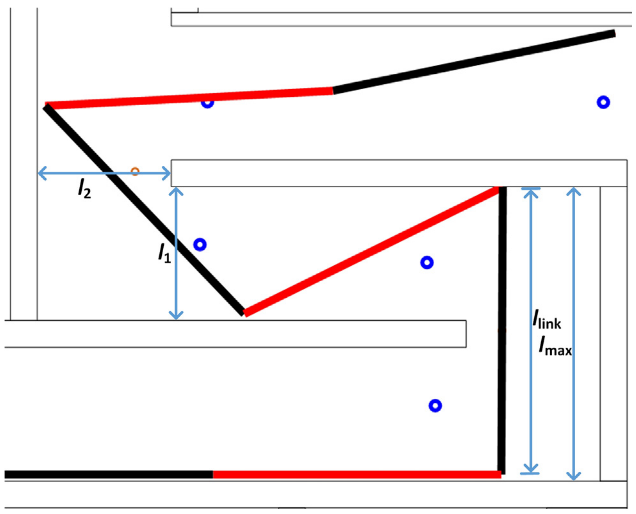


Fig. 19. Benchmark environment.

proceed. The collision-free configuration of each link is calculated individually.

Moreover, it is easily extensible i.e., adding new features to the method has minimum impact to the features of the method. Since the method's basic structure consists of a number of "if-else" statements, adding new feature slightly increases the number of choices.

5.4. Comparison with other methods

5.4.1. Long link lengths

Two exemplary papers [23,32] have been selected to compare the performance of our method. We have drawn conclusion that the index values of the former and latter methods are about 0.25 and 0.24, respectively. Tight maneuvering wasn't considered in these papers because the link lengths are relatively short with respect to the curves of the paths.

If different manipulators are to perform the same task, the one with higher index value would rather be preferred since it accomplishes the task using fewer number of links, thus actuators. To the best of our knowledge, the maneuvering ability of the method is far beyond that of other approaches presented in the literature.

5.5. Short link lengths

A simple path following algorithm has been developed in 3D to compare with other algorithms. The method keeps the link ends on the path points while advancing to the goal point.

The method presented in this paper currently works in 2D. However, it has been developed 3D version in mind. The work for 3D version of the method is under development. Some of the works has already been done such as potential field, obstacles and path implemented in 3D environment for the manipulator.

The method works for a large range from quite short link lengths to quite long link lengths. Long link lengths have already been dealt with above. Using the 3D basis, a comparison with other methods will be performed in the case of relatively short link lengths compared to the curves of the path. Fig. 20 shows a manipulator 182 links whose lengths are 8 units. The link are too short with respect to the curves of the path in such a way that the path is entirely covered with the links and the path cannot be seen.

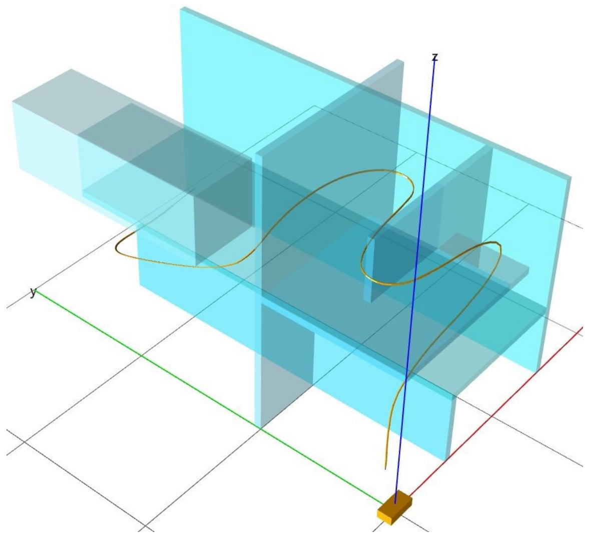


Fig. 20. 3D path following with 182 links whose lengths are 8.

Fig. 21 shows the same path. Nevertheless, this time a manipulator with 49 links whose lengths are 30 units is depicted. Fairly good mapping the links on the path is achieved and from Fig. 21a, it is difficult to distinguish the links from the path. On the other hand, considering Fig. 21b, a closer look reveals the distinction between the links and the path where maneuvering is required.

In Fig. 22, again the same path is followed by a manipulator with 15 links whose lengths are 90 units. As seen from the figure, collision with obstacles is unavoidable. Despite the fact that different mapping algorithms may improve the situation slightly, it does not solve the inherit problem of path following discussed in this paper.

5.5.1. Commercial products

There are commercialized examples of snake robots. The companies coming into prominence among these examples, which are not many but rapidly increasing, are the companies of OC Robotics, HiBot, Medrobotics, Hirose Fukushima Lab ACM and Sintef. These companies produce different types of snake robots, not very similar to each other. Among them, OC Robotics produces the most suitable commercial snake robots enabling us to make comparisons in terms of the maneuvering ability. The maneuvering ability of our path-planning algorithm is far beyond those of their snake robots.

Although the robotic technology in industry has been improved in recent years, even the most advanced industrial robots, including ABB YUMI, KUKA iiwa, and Rethink Baxter/Sawyer, have still the lack of ability to be programmed automatically. These robots are exposed to time-consuming and difficult teaching process by the operators about how to plan the path considering the obstacles [39].

5.6. Implementation

There are two main features of the method. The first one is that it has extremely high manipulability. The second one is that the number of links of the manipulator is only limited to the processing power of an ordinary computer without performance deterioration. The method is practical simply because it can be used with any number of links. Although mechanical design with a large number of links is not an easy task, the method encour-

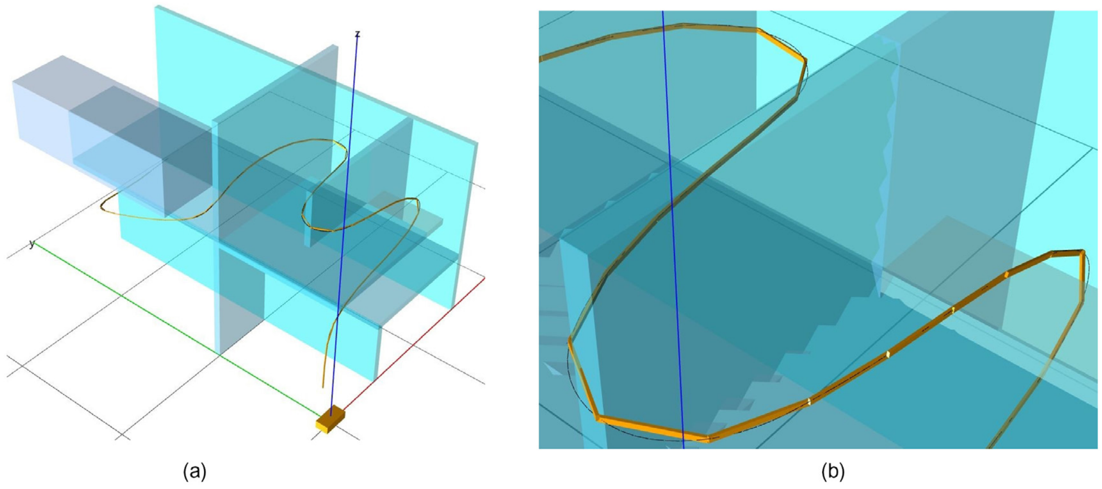


Fig. 21. a-b 3D path following with 49 links whose lengths are 30.

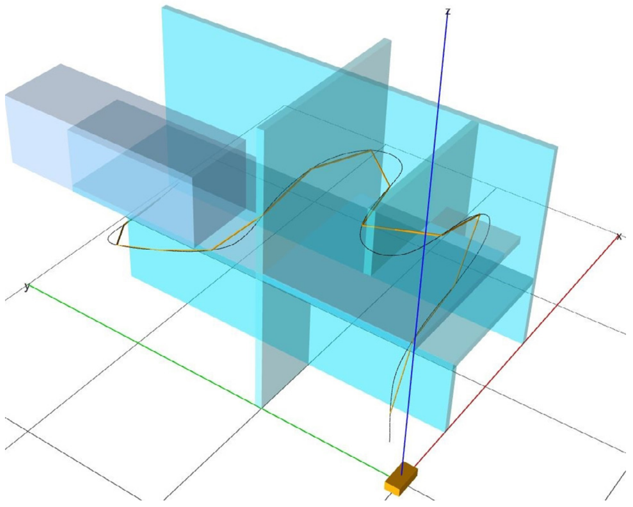


Fig. 22. 3D path following with 15 links whose lengths are 90.

ages designers to face such design challenges. No one can say that it cannot be built. What is the point of attempting to design such a huge manipulator if you know that there is no method to drive it?

Real-time applications are split into two main categories as soft and hard real-time. The violation of deadlines could be tolerated in soft real-time applications. However, hard real-time applications dictate strict deadlines [38]. The category which we are concerned with in this paper is the soft real-time applications. Based on the environment shown in Fig. 17a, 45 motor angles are determined at each configuration. Assuming that hyper-redundant manipulator reaches the goal through obstacles in 20 s in low-level control (hardware), the high level control (our algorithm) only needs 2 s for the whole operation. It means that whenever the manipulator calls for a set of angle values, the algorithm delivers. In other words, the performance of the algorithm is "*sufficiently quickly*", therefore it is soft real-time and suitable for time-critical applications. Since the algorithm make decisions based on various "if" statements for different paths, we cannot put an upper limit for the execution time of each new configuration.

6. Conclusions

We have presented a simple, yet robust approach for path planning of redundant/hyper redundant manipulators in real-time within confined spaces cluttered with obstacles. The algorithm which geometrically plans the path of whole arm with discrete links has been proved to employ the maneuvering space more efficient than its counterparts based on a benchmark scheme and 3D path following examples. A predetermined discretized path for point robots to the goal point gives all the information for navigation of the manipulator. The collision-free configuration of each link is calculated individually. If it is physically possible, the employment of maneuvering space is maximized by making the links almost pinch the walls of the convoluted terrain. In this context, the maneuvering ability of the method is far beyond those of others proposed in the literature. Therefore, for the same path, it lessens the number of links and actuators needed to perform a specific task. Fully utilizing maneuvering space drives down the costs and computational expenses. Although it maneuvers with minimum number of links, the performance of the method is independent of the number of links, but only limited with the power of the computer. The exemplary computer simulations verify the effectiveness of the method while showcasing the notable advantage of proposed method over its counterparts.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.jestch.2020.07.002.

References

[1] E.S. Conkur, R. Buckingham, Clarifying the definition of redundancy as used in robotics, Robotica 15 (1997) 583–586, https://doi.org/10.1017/S0263574797000672.

[2] P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano, Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy, Int. J. Rob. Res. 10 (1991) 410–425, https://doi.org/10.1177/027836499101000409.

[3] Y. Nakamura, Advanced robotics: redundancy and optimization, Choice Rev. Online. 29 (1991), https://doi.org/10.5860/CHOICE.29-0333.

[4] S. Ma, S. Hirose, H. Yoshinada, Development of a hyper-redundant multijoint manipulator for maintenance of nuclear reactors, Adv. Robot. 9 (1994) 281–300, https://doi.org/10.1163/156855395X00201.

[5] S. Seereeram, J.T. Wen, A global approach to path planning for redundant manipulators, IEEE Trans. Robot. Autom. 11 (1995) 152–160, https://doi.org/10.1109/70.345948.

[6] R.J. Schilling, R. Read, V. Lovass-nagy, G. Walker, Path tracking with the links of a planar hyper- redundant robotic manipulator, J. Robot. Syst. 12 (1995) 189–197, https://doi.org/10.1002/rob.4620120304.

[7] O. Takahashi, R.J. Schilling, Motion planning in a plane using generalized Voronoi diagrams, IEEE Trans. Robot. Autom. 5 (1989) 143–150, https://doi.org/10.1109/70.88035.

[8] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, Proc. – IEEE Int. Conf. Robot. Autom. 5 (1985) 500–505, https://doi.org/10.1109/ROBOT.1985.1087247.

[9] P.K. Das, H.S. Behera, B.K. Panigrahi, Intelligent-based multi-robot path planning inspired by improved classical Q-learning and improved particle swarm optimization with perturbed velocity, Eng. Sci. Technol. an Int. J. 19 (2016) 651–669, https://doi.org/10.1016/j.jestch.2015.09.009.

[10] J.-C. Latombe, Robot Motion Planning, Springer, US, Boston MA, 1991. 10.1007/978-1-4615-4022-9.

[11] E.S. Conkur, Path following algorithm for highly redundant manipulators, Rob. Auton. Syst. 45 (2003) 1–22, https://doi.org/10.1016/S0921-8890(03)00083-6.

[12] L. Tang, L.M. Zhu, X.Y. Zhu, G.Y. Gu, Confined spaces path following for cable-driven snake robots with prediction lookup and interpolation algorithms, Sci. China Technol. Sci. 181 (2019), https://doi.org/10.1007/s11431-019-1440-2.

[13] G.S. Chirikjian, J.W. Burdick, An Obstacle Avoidance Algorithm for Hyper-redundant Manipulators, in: 1990: pp. 625–631. https://doi.org/10.1109/robot.1990.126052.

[14] A. Graham, R. Buckingham, Real-time collision avoidance of manipulators with multiple redundancy, Mechatronics 3 (1993) 89–106, https://doi.org/10.1016/0957-4158(93)90040-9.

[15] A. McLean, S. Cameron, Snake-based path planning for redundant manipulators, Proc. – IEEE Int. Conf. Robot. Autom. (1993) 275–282, https://doi.org/10.1109/robot.1993.292158.

[16] A. McLean, S. Cameron, The virtual springs method: path planning and collision avoidance for redundant manipulators, Int. J. Satell. Commun. 15 (1997) 300–319.

[17] S. Ma, M. Konno, Obstacle avoidance scheme for hyper-redundant manipulators – global motion planning in posture space, Proc. – IEEE Int. Conf. Robot. Autom. (1997) 161–166, https://doi.org/10.1109/robot.1997.620032.

[18] T.-C. Liang, J.-S. Liu, An improved trajectory planner for redundant manipulators in constrained workspace, J. Robot. Syst. 16 (1999) 339–351, https://doi.org/10.1002/(SICI)1097-4563(199906)16:6<339::AID-ROB3>3.0.CO;2-1.

[19] H. Choset, W. Henning, A follow-the-leader approach to serpentine robot motion planning, J. Aerosp. Eng. 12 (1999) 65–73, https://doi.org/10.1061/(ASCE)0893-1321(1999)12:2(65).

[20] J.T. Wunderlich, Simulating a robotic arm in a box: redundant kinematics, path planning, and rapid prototyping for enclosed spaces, Simulation 80 (2004) 301–316, https://doi.org/10.1177/0037549704046338.

[21] D. Reznik, V. Lumelsky, Sensor-based motion planning in three dimensions for a highly redundant snake robot, Adv. Robot. 9 (1994) 255–280, https://doi.org/10.1163/156855395X00193.

[22] P.N. Azariadis, N.A. Aspragathos, Obstacle representation by Bump-surfaces for optimal motion-planning, Rob. Auton. Syst. 51 (2005) 129–150, https://doi.org/10.1016/j.robot.2004.11.001.

[23] M.N. Islam, S. Tamura, Implementation of a New Backtrack Free Path Planning Algorithm for Manipulators, PhD Thesis, 2008.

[24] M.N. Islam, S. Tamura, T. Murata, T. Yanase, Evaluation of a new backtrack free path planning algorithm for manipulators, IEEJ Trans. Electron. Inf. Syst. 128 (2008) 1293–1302, https://doi.org/10.1541/ieejeiss.128.1293.

[25] L.A. Burhanuddin, M.N. Islam, S.M. Yusof, Evaluation of Collision Avoidance path planning Algorithm, in: Int. Conf. Res. Innov. Inf. Syst. ICRIIS, IEEE, 2013: pp. 360–365. https://doi.org/10.1109/ICRIIS.2013.6716736.

[26] Y. Miao, F. Gao, Y. Zhang, Gait fitting for snake robots with binary actuators, Sci. China Technol. Sci. 57 (2014) 181–191, https://doi.org/10.1007/s11431-013-5405-0.

[27] A. Jamali, M.R. Khan, M.S. Osman, M.M. Rahman, M.F. Ashari, M.S. Jamaludin, E. Junaidi, Collision free control of variable length hyper redundant robot manipulator, Appl. Mech. Mater. 541–542 (2014) 1107–1114, https://doi.org/10.4028/www.scientific.net/AMM.541-542.1107.

[28] T. Collins, W.-M. Shen, PASO: An Integrated, Scalable PSO-based Optimization Framework for Hyper-Redundant Manipulator Path Planning and Inverse Kinematics, in: 2015. https://www.isi.edu/robots/prl/collins2016-ISI-TR-697.pdf.

[29] S. Tappe, J. Pohlmann, J. Kotlarski, T. Ortmaier, Towards a follow-the-leader control for a binary actuated hyper-redundant manipulator, IEEE Int. Conf. Intell. Robot. Syst., IEEE (2015) 3195–3201, https://doi.org/10.1109/IROS.2015.7353820.

[30] S. Tappe, J. Pohlmann, J. Kotlarski, T. Ortmaier, Optimization Strategies for Task Specific Path-Following Capabilities of a Binary Actuated Snake-Like Robot using Follow-The-Leader Control, in: IEEE/ASME Int. Conf. Adv. Intell. Mechatronics, AIM, 2017: pp. 1574–1581. https://doi.org/10.1109/AIM.2017.8014243.

[31] L. Tang, J. Wang, Y. Zheng, G. Gu, L. Zhu, X. Zhu, Design of a cable-driven hyper-redundant robot with experimental validation, Int. J. Adv. Robot. Syst. 14 (2017) 1–12, https://doi.org/10.1177/1729881417734458.

[32] L. Tang, L.M. Zhu, X. Zhu, G. Gu, A Serpentine Curve Based Motion Planning Method for Cable-Driven Snake Robots, in: Proc. 2018 25th Int. Conf. Mechatronics Mach. Vis. Pract. M2VIP 2018, 2019: pp. 5–10. https://doi.org/10.1109/M2VIP.2018.8600874.

[33] J. De Maeyer, M. Versteyhe, E. Demeester, Sampling-based Tube Following for Redundant, Planar Robotic Manipulators, in: IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA, IEEE, 2018: pp. 752–758. https://doi.org/10.1109/ETFA.2018.8502490.

[34] H. Xie, C. Wang, S. Li, L. Hu, H. Yang, A geometric approach for follow-the-leader motion of serpentine manipulator, Int. J. Adv. Robot. Syst. 16 (2019) 1–18, https://doi.org/10.1177/1729881419874638.

[35] E.S. Conkur, R. Buckingham, Manoeuvring highly redundant manipulators, Robotica 15 (1997) 435–447, https://doi.org/10.1017/S0263574797000532.

[36] E.S. Conkur, R. Buckingham, A. Harrison, The beam analysis algorithm for path planning for redundant manipulators, Mechatronics 15 (2005) 67–94, https://doi.org/10.1016/j.mechatronics.2004.06.009.

[37] E.S. Conkur, Path planning using potential fields for highly redundant manipulators, Rob. Auton. Syst. 52 (2005) 209–228, https://doi.org/10.1016/j.robot.2005.03.005.

[38] B. Srinivasan, S. Pather, R. Hill, F. Ansari, D. Niehaus, A firm real-time system implementation using commercial off-the-shelf hardware and free software, in: Fourth IEEE Real-Time Technology and Applications Symposium Proceedings, 1998: pp. 112–119, .

[39] M. Stenmark, E.A. Topp, From demonstrations to skills for high-level programming of industrial robots, in, AAAI Fall Symposium Series 2016 (2016) 75–78.