# Trajectory Planning and Obstacle Avoidance for Hyper-Redundant Serial Robots

**Midhun S. Menon**
Department of Mechanical Engineering,
Indian Institute of Science,
Bangalore 560 012, India
e-mail: midhun.sreekumar@gmail.com

**V. C. Ravi**
Centre for AI & Robotics,
Bangalore 560 093, India
e-mail: vc_ravi@cair.drdo.in

**Ashitava Ghosal**[1]
Department of Mechanical Engineering,
Indian Institute of Science,
Bangalore 560 012, India
e-mail: asitava@mecheng.iisc.ernet.in

*Hyper-redundant snakelike serial robots are of great interest due to their application in search and rescue during disaster relief in highly cluttered environments and recently in the field of medical robotics. A key feature of these robots is the presence of a large number of redundant actuated joints and the associated well-known challenge of motion planning. This problem is even more acute in the presence of obstacles. Obstacle avoidance for point bodies, nonredundant serial robots with a few links and joints, and wheeled mobile robots has been extensively studied, and several mature implementations are available. However, obstacle avoidance for hyper-redundant snakelike robots and other extended articulated bodies is less studied and is still evolving. This paper presents a novel optimization algorithm, derived using calculus of variation, for the motion planning of a hyper-redundant robot where the motion of one end (head) is an arbitrary desired path. The algorithm computes the motion of all the joints in the hyper-redundant robot in a way such that all its links avoid all obstacles present in the environment. The algorithm is purely geometric in nature, and it is shown that the motion in free space and in the vicinity of obstacles appears to be more natural. The paper presents the general theoretical development and numerical simulations results. It also presents validating results from experiments with a 12-degree-of-freedom (DOF) planar hyper-redundant robot moving in a known obstacle field.* [DOI: 10.1115/1.4036571]

*Keywords: hyper-redundant robots, motion planning, obstacle avoidance, optimization, 12-link planar robot*

## 1 Introduction

In an earthquake or similar disaster situations with debris and narrow passages, the application of hyper-redundant snakelike robots is being increasingly explored for search and rescue as humans and other robots are less effective in these environments [1]. Likewise, in medical robotics, tools and techniques from research in hyper-redundant robots are being increasingly used in developing advanced actuated endoscopes and virtual surgery simulators where tasks such as suturing or motion of flexible arteries need to be shown in a realistic manner [2]. Finally, in the animation of flexible objects such as a string, hair, and flexible hoses, the flexible object is discretized into a large number of rigid objects and, again, techniques used for motion planning in hyper-redundant robots are used [3]. This paper deals with the development and experimental validation of an algorithm for realistic and efficient motion planning of hyper-redundant serial robots in free space as well as in the presence of obstacles where these obstacles are avoided by the entire robot.

**1.1 Problem Statement.** Given an $n$-link hyper-redundant snakelike robot ($n \gg 6$ in three-dimensional (3D) and $n \gg 3$ in two-dimensional (2D)) and a desired path for the head to follow through a field of smooth, implicitly defined obstacles, plan the motion of the entire robot with obstacle avoidance.

**1.2 Comparison With Existing Algorithms.** The classical problem of obstacle avoidance for point bodies has been studied in depth, and many solutions have been proposed. The review paper by Hwang and Ahuja [4] contains a good overview of the methods and tools available for this problem. Some of the methods include mapping obstacle information onto a precomputed space [5,6], using graph theoretic constructions [7], potential field techniques [8–10], Voronoi diagrams [11], artificial neural networks [12], polyhedral interference detection based on computational geometry [13], reinforcement learning algorithms [14], dynamic programming [15], and optimal control [16–19]. However, these methods do not explicitly address the extended and articulated nature of the hyper-redundant snake robot's physical structure.

For general hyper-redundant manipulators, the standard approaches for motion planning are based on the pseudo-inverse of the Jacobian and its variants, including the task-augmented Jacobian and the extended Jacobian methods [20,21]. Another approach to motion planning involves mapping the obstacles into the configuration space of the robot and then determining a feasible path through configuration space to accomplish the task at hand [22–24]. A discrete modal summation approach has been used in Ref. [25] to constrain the manipulator into obstacle free zones called "tunnels." A follow-the-leader approach for obstacle avoidance has been proposed in Ref. [26]. These methods essentially divide the motion planning problem into two distinct phases—one in free space and one in the vicinity of obstacles. Artificial neural networks [27] and optimal control [28] strategies have also been suggested for motion planning.

The primary drawback of some of these methods is that a significant amount of engineering and algorithmic intuition is required to formulate the algorithm for motion planning. For example, in the Jacobian-based methods, it has to be ensured that algorithmic singularities are either not encountered or at least addressed explicitly, which limits the applicability to well-known environments. In the modal approach, the choice of modal functions is a nontrivial task, where several sets of modes may need to be defined to span the workspace and a mode switching mechanism implemented to ensure smooth transitions between modes. Some of these methods are also computationally very expensive. For example, configuration space methods are generally not computationally feasible for hyper-redundant robots due to the higher

[1]Corresponding author.

Manuscript received July 22, 2016; final manuscript received March 25, 2017; published online May 15, 2017. Assoc. Editor: Satyandra K. Gupta.

dimensionality of the configuration space. In general, except for the follow-the-leader approach, none of the methods provide intuitive solutions from the perspective of a human machine interface, which can be a serious drawback during, for example, teleoperation by a human operator. The follow-the-leader approach, while being effective in the immediate vicinity of an obstacle, is inefficient in relatively open spaces since it does not minimize the velocity of the links.

An interesting approach, called obstacle aided locomotion, has been used in Refs. [29] and [30], wherein the obstacles are used to generate reaction forces for locomotion, thus mimicking natural snakes, and in a sense obviating the problem of obstacle avoidance. Some of the recent papers also deal with the problem using rapidly exploring random tree (RRT) algorithm [31] and hybrid of RRT and motion primitives (MP) algorithm [32]. However, these algorithms are partly stochastic in nature and cannot guarantee asymptotic optimality. Moreover, it also assumes that the final pose of the robot is known a priori, which need not be the case always.

In Ref. [3], the authors have proposed an intuitive and computationally inexpensive motion planner based on the tractrix curve [33] for free space motion. It is shown that the algorithm has the interesting property of attenuating motion of the links as one traverses from the head to the tail, and, as a result, the motion appears more natural. In this follow up paper, obstacle avoidance is incorporated using a constrained Lagrangian formulation. The algorithm is computationally efficient as it breaks down the obstacle avoidance problem for the $n$-degree-of-freedom (DOF) hyper-redundant manipulator to $n$ obstacle avoidance problems for 1DOF rigid links. It is also shown that in free space, the motion is along the link, and in the vicinity of the obstacle the link moves along the local normal to the obstacle surface.

The paper is organized as follows. Section 2 explains the constrained Lagrangian formulation of the obstacle avoidance problem for an extended body. Section 3 presents numerical simulation results illustrating the motion of a hyper-redundant robot with 12-links in a plane and another hyper-redundant robot with 40 links in 3D space. In Sec. 4, a prototype 12DOF hyper-redundant robot is described, and experimental results are presented and discussed. Finally, the conclusions and scope for further enhancement of this work are presented in Sec. 5.

## 2 Constrained Lagrangian Formulation

In this section, we present the details of the optimization-based approach. We first present in brief the formulation and a general result for an arbitrary curve, one end of which is given a prescribed motion in free space. Next, the results for a single straight rigid object are given, and it is shown that the motion of the distal end traces a classical curve, called the tractrix, when the proximal end (head) is moved along a straight line (see Ref. [3]). To incorporate the obstacles in the optimization-based approach, additional constraints are added, and this requires the boundary of the obstacles to be modeled by smooth differentiable functions. In this work, we model obstacles as superellipses or superellipsoids, and this is described in this section. Finally, the motion planning problem for an $n$ -link hyper-redundant robot is formulated, and general theoretical results are presented.

### 2.1 Formulation for a Curve.

Consider a planar curve of length $L$, parametrized by its arc length $s$, with one of its ends given a prescribed motion. The objective is to find the motion of the curve which minimizes a velocity-based metric subject to the constraint that the length of the curve is preserved.

The curve at any instant $t$ can be written in terms of a spatio-temporal parametrization as

$$C: (T_x(t)+x(s,t), T_y(t)+y(s,t)) \tag{1}$$

The terms $(x(s,t), y(s,t))$ define the curve configuration relative to the perturbed tip, i.e., the curve configuration when viewed from a moving coordinate system attached to the perturbed tip. Note that, in this proposed parametrization, the functions $x(0,t)=0$ and $y(0,t)=0$ at any instant $t$. This is due to the fact that at $s=0$ (leading end), the absolute displacements are completely specified by the predefined functions $(T_x(t), T_y(t))$. For the infinitesimal displacement from time $t$ to $t+\Delta t$, the "distance/velocity" [3] between the two curves can be defined as

$$L_2: \int_0^L \sqrt{\left(\frac{dT_x}{dt}+\frac{\partial x(s,t)}{\partial t}\right)^2+\left(\frac{dT_y}{dt}+\frac{\partial y(s,t)}{\partial t}\right)^2}\,ds \tag{2}$$

The above represents an $\boldsymbol{L}^2$ metric based on the velocity.

We also wish to impose the constraint that the length of the curve is preserved during the motion, which we instantiate by requiring that the sum of all changes in segment lengths is zero, and this can be written as

$$\int_0^L \left(\sqrt{\left(\frac{\partial x(s,t)}{\partial s}\right)^2+\left(\frac{\partial y(s,t)}{\partial s}\right)^2}-1\right)ds=0 \tag{3}$$

It may be noted that the total length of the curve is preserved, and there can be local expansion or contraction.

The above problem is then posed as a constrained Lagrangian optimization problem as follows:

$$\begin{aligned}&\underset{x(s,t),y(s,t)}{\text{Min}}\ I:\ \int_0^L\int_0^T \sqrt{\left(\frac{dT_x}{dt}+\frac{\partial x}{\partial t}\right)^2+\left(\frac{dT_y}{dt}+\frac{\partial y}{\partial t}\right)^2}\,dt\,ds\\ &\text{Subject to}\\ &\Lambda(t):\ \ A=\int_0^L\left(\sqrt{\left(\frac{\partial x}{\partial s}\right)^2+\left(\frac{\partial y}{\partial s}\right)^2}-1\right)ds=0\\ &\text{Data}: x(s,0), y(s,0), T_x(t), T_y(t), x(0,t)=0, y(0,t)=0\end{aligned} \tag{4}$$

where $\Lambda(t)$ is the Lagrangian multiplier corresponding to the length-preserving constraint. Following the calculus of variation approach, the Lagrangian $\mathcal{L}$ for the above optimization is given as $I+\Lambda(t)A$. Writing out the corresponding Euler–Lagrange equations [34,35] and solving give the following expression:

$$\frac{\frac{\partial}{\partial s}y(s,t)}{\frac{\partial}{\partial s}x(s,t)}=\frac{\frac{d}{dt}T_y(t)+\frac{\partial}{\partial t}y(s,t)}{\frac{d}{dt}T_x(t)+\frac{\partial}{\partial t}x(s,t)}=\frac{\frac{\partial}{\partial t}(T_y(t)+y(s,t))}{\frac{\partial}{\partial t}(T_x(t)+x(s,t))} \tag{5}$$

The extreme left-hand side of Eq. (5) is the spatial derivative or the slope at a given $s$ and $t$, and the far right-hand side is the temporal derivative or the velocity vector for a given $s$ and $t$. This implies that for the curve, the $\boldsymbol{L}^2$ metric as defined in Eq. (2) is minimized if the velocity vector at any $(s, t)$ is along the instantaneous tangent to the curve at that point. In addition, during this minimizing motion, the total arc length of the curve is preserved.

### 2.2 Formulation for a Rigid Link.

For a single straight rigid link, the velocity minimizing motion is *along* the link. For simplicity, if the input perturbation is chosen to point along the $X$ axis and the perturbed end lies on the $X$ axis initially, the perturbation functions in Eq. (5) take the form

$$\frac{d}{dt}T_x(t)=1,\quad T_y(t)=0 \tag{6}$$

Since the curve $(x(s,t), y(s,t))$ is a straight line, we have

$$x(s,t)=a(t)s,\quad y(s,t)=b(t)s \tag{7}$$

and since the length of the curve $L$ is preserved, we get

$$\sqrt{(aL-0)^2+(bL-0)^2}=L \Rightarrow a^2+b^2=1 \tag{8}$$

Furthermore, by assuming that the straight rigid link is vertical at $t=0$, we get $a(0)=0$ and $b(0)=1$. Substituting these in Eq. (5) and simplifying, we get the equation for curve traced by distal end as follows:

$$x(L,p)=p-L\tanh\left(\frac{p}{L}\right), \quad y(L,p)=L\,\mathrm{sech}\left(\frac{p}{L}\right) \tag{9}$$

The above parametric equations represent the classical curve called tractrix [36].

**2.3 Motion Planning for a Hyper-Redundant Robot.** Consider an $n$-link hyper-redundant robot as shown schematically in Fig. 1. For the prescribed motion $(T_x(t), T_y(t))$ of the leading end of the first link (point 1), Eq. (9) can be used to obtain the motion of the point 2. It may be noted that the point 2 is also the leading end of the second link, and using Eq. (9), together with appropriate coordinate transformations, we can obtain the location of point 3, which is also the leading end of the third link. Proceeding in a similar manner, we can obtain the motion of all points $4,5,\ldots,j,\ldots,(n-1),n$. Once the locations of all the points are known, we can obtain the vectors between points $i-1$, $i$, and $i+1$ and, by taking a dot product, obtain the rotations at all the $n-1$ joints. This is the tractrix-based algorithm. More details of the tractrix-based resolution of redundancy, such as extension to spatial 3D motion, can be obtained from Refs. [2], [3], and [33]. We list some of the main features of this resolution scheme.

(1) As shown in above mentioned references, the motion of the distal end of a link is less than or equal to the motion given at the near end. As a result, the velocity attenuates as one travels down the serial chain away from the end where the prescribed motion is given.
(2) As shown in Ref. [3], if the direction of the prescribed motion is not changed, as time progresses, the velocity vector of the distal end aligns with the direction of the prescribed motion.
(3) The tractrix-based algorithm has a complexity of $\mathcal{O}(n)$, where $n$ is the number of links.

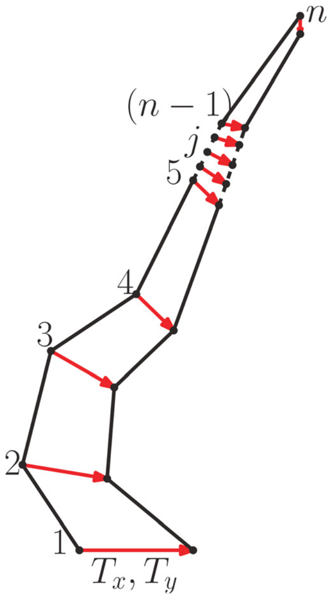


**Fig. 1 Motion planning for a generic hyper-redundant robot**

The first two features impart realism to the motion of an extended flexible object discretized by $n$ rigid links or in a hyper-redundant serial robot, and the last feature results in real-time and efficient motion planning and rendering. We extend the tractrix-based algorithm to motion planning in the presence of obstacles in this work. The obstacles are added as constraints in the optimization, and in the following, we present an approach to mathematically model obstacles as smooth and differentiable functions, thereby making them amenable to the optimization-based approach.

**2.4 Obstacle Modeling.** Given any sphere (or it's topological equivalents) $\mathcal{C} \in \mathbb{R}^n$, $n>1$, by Jordan–Brouwer theorem [37], $\mathbb{R}^n \setminus \mathcal{C}$ has precisely two components, *interior* ($\mathcal{I}$) and *exterior* ($\mathcal{E}$) with boundary $\mathcal{C}$. In 3D space, any topological sphere partitions the spatial points into interior set ($\mathcal{I}$), exterior set ($\mathcal{E}$), and boundary set ($\mathcal{C}$) based on the value of the map $f(\boldsymbol{P}): \mathbb{R}^3 \to \mathbb{R}$, whose zero-level set is the implicit representation of the obstacle boundary (i.e., $f(\boldsymbol{P})=0$). Mathematically, we have

$$\mathcal{E}=\{\boldsymbol{P}|f(\boldsymbol{P})>0\}, \mathcal{I}=\{\boldsymbol{P}|f(\boldsymbol{P})<0\}, \mathcal{C}=\{\boldsymbol{P}|f(\boldsymbol{P})=0\} \tag{10}$$

Smooth obstacles with differentiable implicit boundary representations can be incorporated as constraints in the optimization problem and solved using classical gradient-based algorithms. In case multiple obstacles $\mathcal{O}_j$, $1\le j\le m$ are present, each with an exterior $\mathcal{E}_j$, the intersection of all the individual exteriors gives the permissible space for motion planning, namely, $\mathcal{E}=\cap_{j=1}^m \mathcal{E}_j$. Hence, the statement of the optimization problem, including obstacle avoidance, is as follows:

$$\begin{aligned}
&\underset{\boldsymbol{P}_{i+1}(t_2))}{\text{Min}}\ I: (\Delta(T_x+x_{i+1}))^2(\Delta(T_y+y_{i+1}))^2\\
&\text{Subject to}\\
&\lambda_i:\ \sqrt{(x_{i+1}(t_2)-x_i(t_2))^2+(y_{i+1}(t_2)-y_i(t_2))^2}=L\\
&\beta_j:\ f_j(\boldsymbol{P})>0\ \forall\ j\in[1,m]\\
&\text{Data}:\ \boldsymbol{P}_i(t_1), \boldsymbol{P}_i(t_2), T_x(t_1), T_y(t_1), T_x(t_2), T_y(t_2), \boldsymbol{P}_1(t_1)=(0,0)^T\\
&\forall i\in[1,n-1]\ \&\ \forall\ t_1\in[0\ldots T],\ t_2=t_1+\Delta t
\end{aligned} \tag{11}$$

In this paper, the obstacle shapes are restricted to smooth and differentiable superellipsoids (see Refs. [38, Chap. 18] and [39]) of the form

$$\left(\left(\frac{x}{a_1}\right)^{\frac{2}{\varepsilon_1}}+\left(\frac{y}{a_2}\right)^{\frac{2}{\varepsilon_1}}\right)^{\frac{\varepsilon_1}{\varepsilon_2}}+\left(\frac{z}{a_3}\right)^{\frac{2}{\varepsilon_2}}=1 \tag{12}$$
$$0\le\varepsilon_1\le1,\ 0\le\varepsilon_2\le1$$

For various values of the parameters $\varepsilon_1, \varepsilon_2, a_1, a_2, a_3$, Eq. (12) generates a family of shapes which includes circles, ellipses, rectangles, cylinders, cuboids, cubes, ellipsoids, spheres, etc. Some of these with their associated parameters are shown in Figs. 2 and 3. Note that $\varepsilon_1=p_2/q_2$, $\varepsilon_2=p_1/q_1$. Obstacles in the environment can be modeled using a combination or union of one or more of these shapes. In simulations and experiments described in Secs. 3 and 4, we have grown the obstacles using Minkowski sum [22] based on the link length of the robot. As already known in geometry, the Minkowski sum (also known as dilation) $C$ of two sets of position vectors $A$ and $B$ in Euclidean space is formed by pointwise sum of each vector in $A$ to each vector in $B$, i.e., the set

$$C=A+B=\{\mathbf{a}+\mathbf{b}|\mathbf{a}\in A, \mathbf{b}\in B\} \tag{13}$$

In our case, $A$ is the set of position vectors of all points lying on the link for all possible orientations of the link relative to the trailing end, and $B$ is the union of interior and boundary of the obstacle being considered. For a link of length $L$ and angle $\theta$, these may be expressed as

$$A = \{(l\cos\theta, l\sin\theta) | 0 \le l \le L, 0 \le \theta \le 2\pi\}$$
$$B = \{\boldsymbol{P} | f(\boldsymbol{P}) \le 0\} \tag{14}$$

**2.5 Optimization Formulation for a Hyper-Redundant Robot.** As shown in Ref. [3], the optimization of the articulated chain (in free space) is equivalent to the iteration of the tractrix curve equations to a single link at a time. Hence, we consider the case of a single link and, without loss of generality, consider a single obstacle with boundary represented by implicit function $f(x,y) = 0$. These assumptions simplify the 2D variational problem (4) to

$$\begin{aligned}
&\underset{x(t),y(t)}{\text{Min}}\ I\ :\ \int_0^T \sqrt{(\dot{T}_x + \dot{x})^2 + (\dot{T}_y + \dot{y})^2}\, dt \\
&\text{Subject to} \\
&\Lambda(t):\ \ A = x^2 + y^2 - L^2 = 0 \\
&\Omega(t) \ge 0:\ \ B = f(x,y) \ge 0 \\
&\text{Data}:\ \ x(0), y(0), T_x(t), T_y(t)
\end{aligned} \tag{15}$$

Applying the Euler–Lagrange equations on the augmented Lagrangian $\mathcal{L} = I + \Lambda(t)A - \Omega(t)B$ and simplifying, we get

$$\dot{T}_x + \dot{x} = R\left(\Omega(t)\frac{\partial f}{\partial y} - 2\Lambda(t)x\right) \tag{16a}$$

$$\dot{T}_y + \dot{y} = R\left(\Omega(t)\frac{\partial f}{\partial x} - 2\Lambda(t)y\right) \tag{16b}$$

where $R$ is the radius of curvature of the trailing end trajectory, and $\Omega(t)$ and $\Lambda(t)$ are the Lagrange multipliers. From the above equations, we can see that the velocity spans a half-space defined by the linear combination of obstacle outward normals and a vector along the link. This is illustrated in Fig. 4. In the absence of obstacles, the velocity of the trailing end is along the link—this is the classical tractrix curve solution. In the presence of obstacles, the velocity of the trailing end is not along the link but lies in the half-space thereby ensuring obstacle avoidance. The exact direction of the velocity depends on the relative magnitudes of the Lagrange multipliers but the magnitude of the velocity is higher in areas where the trajectory direction changes rapidly (small $R$) and viceversa. The overall workflow of the algorithm has been summarized as a flowchart in Fig. 5.

## 3 Numerical Simulation Results

In this section, we present the results of numerical simulation[2] carried out for a chosen hyper-redundant robot. The leading end of the hyper-redundant robot is moved along a generic trajectory in two- and three-dimensional space in the presence of obstacles. In all the simulations, without loss of generality, the initial configuration of the robot is chosen to be a straight line.

In the 2D simulation, the hyper-redundant robot consists of 12 rigid segments connected by revolute joints yielding a system with 12 degrees-of-freedom. The leading end is moved along an arbitrarily chosen trajectory (shown in Fig. 6). As mentioned earlier, the obstacles are modeled as superellipses or as combination of superellipses. Along the path, six arbitrary snapshot locations are chosen. The initial configuration (at snapshot 1) of the hyper-redundant robot is shown in black color. The configuration of the hyper-redundant robot at each of the six snapshot locations of the leading end is shown in Fig. 7. As seen here, the obstacles are avoided, and motion is minimized as one moves away from the leading end of the flexible object. In other words, tractrix motion is followed in obstacle free spaces, and the algorithm automatically switches to obstacle avoidance once the objects are encountered. The plots of three joint angles (head, tail, and middle link) for the 2D simulation are shown in Fig. 8(*a*). The time shifted behavior of the three plots (indicated at the labeled peaks) indicates the delay for each moving link to encounter a spatially fixed obstacle and avoid it. The jump in joint angle which results in obstacle avoidance is due to the positive spike in Lagrange multipliers in the vector $\Omega(t)$ which acts like a kinematic repulsion similar to the virtual forces in an artificial potential-based method. These spikes are shown for all links for obstacle 2 (elliptical) in Fig. 8(*b*). Clearly, on correlating Figs. 8(*a*) and 8(*b*), it is seen that in the vicinity of obstacle, the Lagrange multiplier $\Omega$ is nonzero,

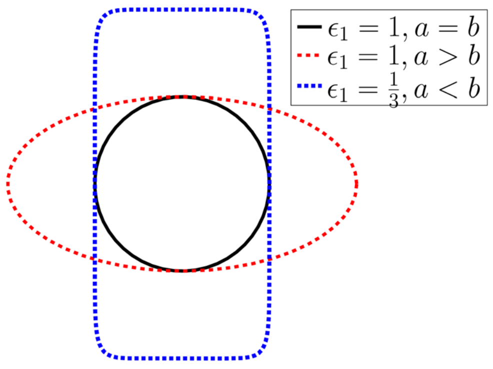


**Fig. 2 Analytic 2D obstacles generated as superellipses**

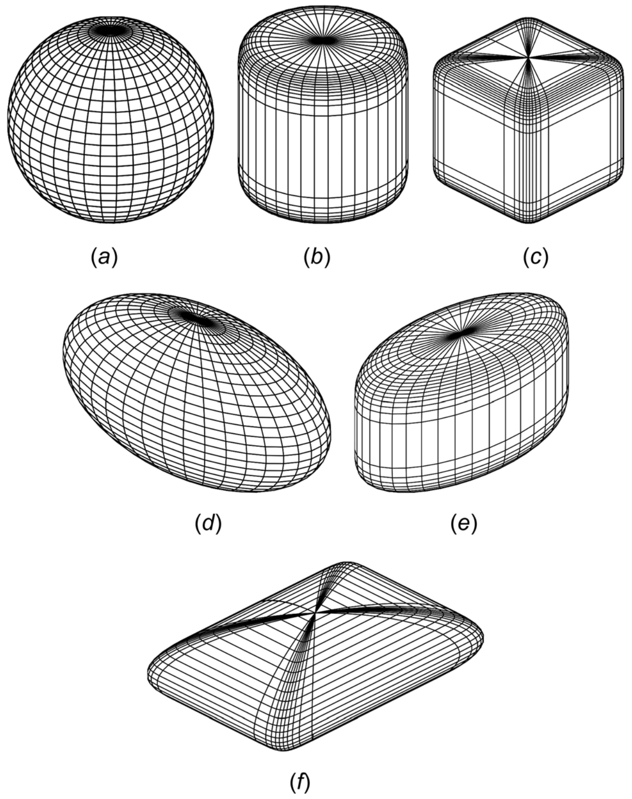


**Fig. 3 Superellipsoids ((*a*)–(*c*) are scaled uniformly and (*d*)–(*f*) are scaled nonuniformly): (*a*) $(p_1/q_1) = 1, (p_2/q_2) = 1$, (*b*) $(p_1/q_1) = (1/5), (p_2/q_2) = 1$, (*c*) $(p_1/q_1) = (1/5), (p_2/q_2) = (1/5)$, (*d*) $p_1/q_1 = 1, p_2/q_2 = 1$, (*e*) $(p_1/q_1) = (1/3), (p_2/q_2) = 1$, and (*f*) $(p_1/q_1) = 1, (p_2/q_2) = 1/7$**

[2]All simulations were done using the *fmincon* routine in MATLAB [40] on a Pentium quad core PC with 16 GB RAM running Linux OS.

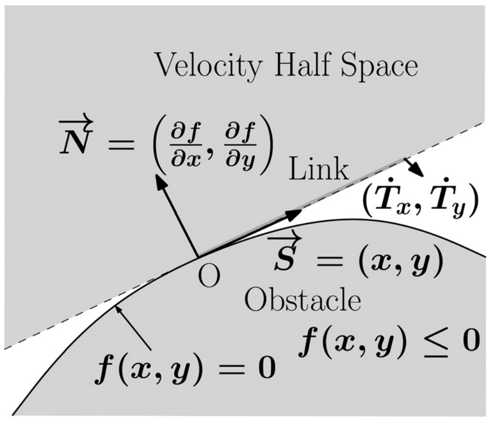

Velocity Half Space
$\overrightarrow{N} = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$
Link
$(\dot{T}_x, \dot{T}_y)$
O
$\overrightarrow{S} = (x, y)$
Obstacle
$f(x,y) \le 0$
$f(x,y) = 0$

Fig. 4 Span of calculated link velocity solutions

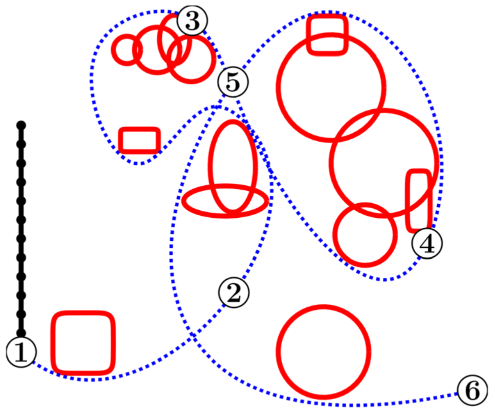


Fig. 6 Trajectory for 2D simulation with snapshot locations and initial configuration of the robot

and hence the obstacle constraint is active. This confirms the derived theoretical results in the optimization procedure.

In the second simulation, the motion planning of a hyper-redundant robot is done for an arbitrarily chosen 3D motion of the leading end. Here too, the initial configuration of the object is chosen as a straight line. The hyper-redundant robot is assumed to have 40 rigid segments with two DOF joints connecting consecutive segments. The leading end is subjected to an arbitrarily chosen motion discretized into steps of 0.2 length units, and the total motion is for 400 steps. The trajectory is generated in an obstacle field with seven obstacles modeled as superquadrics (Fig. 9(*a*)). It can be seen that the extended body avoids obstacles optimally by grazing them tangentially (Figs. 9(*b*)–9(*h*)). This demonstrates the efficacy of the algorithm. Finally, Table 1 shows the execution times for the 3D simulation when the number of links in the hyper-redundant robot is increased. It can be observed that the simulation time increases approximately linearly with the number of links ($n$). This is expected since, in free space, the tractrix-based algorithm is of linear complexity and the deviation from the tractrix-based algorithm *only* occurs when the link is near an obstacle.

## 4 Experimental Results

To illustrate the algorithms developed in Sec. 2, we fabricated a serial hyper-redundant robot. The hyper-redundant robot consists of 12-links as shown in Fig. 10(*a*). It has single degree-of-freedom revolute joints connecting each adjacent pair of links. The axes of all joints are parallel to the vertical and thus allowing only planar motion of each link. Each link additionally is

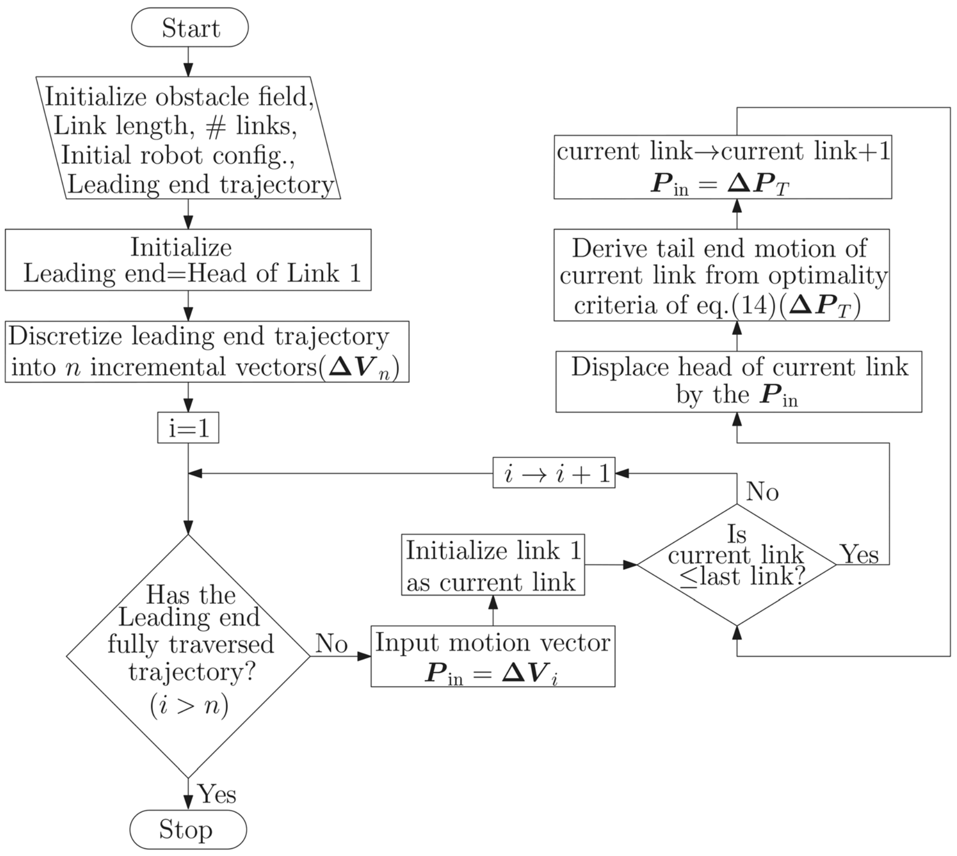

Start
Initialize obstacle field, Link length, # links, Initial robot config., Leading end trajectory
Initialize Leading end=Head of Link 1
Discretize leading end trajectory into $n$ incremental vectors($\boldsymbol{\Delta V}_n$)
i=1
Has the Leading end fully traversed trajectory? ($i > n$)
Yes
Stop
No
Input motion vector $\boldsymbol{P}_{\text{in}} = \boldsymbol{\Delta V}_i$
Initialize link 1 as current link
Is current link ≤last link?
No
$i \to i+1$
Yes
Displace head of current link by the $\boldsymbol{P}_{\text{in}}$
Derive tail end motion of current link from optimality criteria of eq.(14)($\boldsymbol{\Delta P}_T$)
current link→current link+1 $\boldsymbol{P}_{\text{in}} = \boldsymbol{\Delta P}_T$

Fig. 5 Flowchart of motion planning algorithm with obstacle avoidance

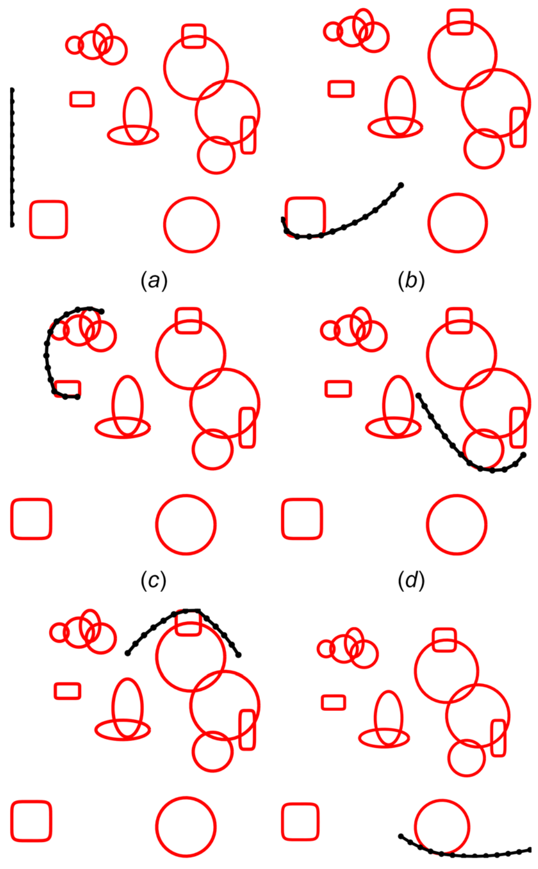


**Fig. 7 Motion snapshots for 2D simulation: (*a*) snapshot 1, (*b*) snapshot 2, (*c*) snapshot 3, (*d*) snapshot 4, (*e*) snapshot 5, and (*f*) snapshot 6**

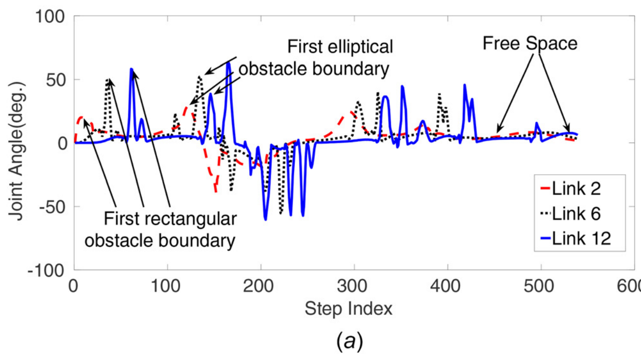
supported on a powered wheel to enable overall forward mobility of the mechanism, and at each instant the position and orientation of the head in the *XY*-plane is specified. Each link consists of a bracket for the two motors, one for the body joint and the other for the wheel motor, as seen in Fig. 10(*b*). The link length, measured from one body joint to the next, is 85 mm. The links were fabricated using a 3D printing machine, and the wheels are mounted alternately on either side of the robot.

The joints are driven by standard Futaba S3003 RC hobby servos, while the wheels are driven by SpringRC SM-S4303R continuous rotation (CR) hobby servos. The servos have a three-wire interface, with one wire each for positive supply, negative return, and command input signal. The servos feature an integrated closed-loop controller to maintain the position/speed according to the command input pulse, thus making a compact actuator ideal for this purpose. On the flip side, the three-wire interface of the servos does not permit higher level monitoring of the motor position or speed, thus reducing the possibility of effective higher level motion control. The servos take a command position in the form of a pulse-width modulated (PWM) pulse with a nominal time period of ~30 ms. For the S3003 servos, a pulse width of ~1 ms corresponds to the $-90$ deg position, and a pulse width of 2 ms corresponds to the $+90$ deg position, with the motor centered at 1.5 ms. For the SM-S4303 servos, the motor is at rest at a commanded pulse width of ~1.5 ms. Higher pulse widths cause a clockwise rotation, while lower pulse widths cause counterclockwise rotation, with the speed varying approximately with pulse width. The CR servos were calibrated to determine the relation between pulse width and motor speed, and motors with near linear pulse-speed relation were used. These motors were distributed along the length of the body to maintain the balance of the robot when it moves along a straight line. A custom designed PIC 18*F*252 microcontroller-based board is used to generate the command pulses for all the 12 joint servomotors. An identical board is used to control the wheel motors. Both the boards also have an RS232 serial interface port to communicate with a PC.

Several experiments were conducted for different choices of obstacle placements and paths. The results of one such experiment are presented below. The experiments were conducted on a planar smooth tiled floor, with known obstacle locations. Obstacles were chosen from among a set of a rectangular object and circular objects of three different sizes. Different numbers, shapes, sizes, and relative locations of the obstacles were chosen for each experiment. The initial orientation of the robot was arbitrarily chosen to be a straight line lying along the *Y*-axis and pointing in the $-Y$ direction. Way-points were manually chosen for the robot to pass between and beyond the obstacles, after inflating the obstacles suitably to account for the robot's width and link length using Minkowski summing operation.

In these experiments, the path of the robot's head and the joint configuration of the robot, as the head moves along this path in discrete steps, are calculated using the algorithm as described in Sec. 2. The sequence of these joint configurations is then fed to the robot through the PC's RS232 interface to the PIC18F252 board controlling the joint servomotors. The wheel kinematics has not been considered in these experiments. The calculated wheel speeds are fed to the robot through the PC's RS232 interface to the PIC18F252 board controlling the wheel servomotors. The joint configurations and wheel speeds are synchronized at each step. The algorithm itself is implemented in MATLAB, while the interface to the PIC18F252 boards is implemented in C, both running under Windows XP on an Intel Xeon workstation with 2 GB of RAM.

Figure 11(*a*) shows a simulated view of the workspace and obstacles, with the calculated path of the robot's head. Figure 11(*b*) shows a plot of three joint angles—joints 1 (head), 6



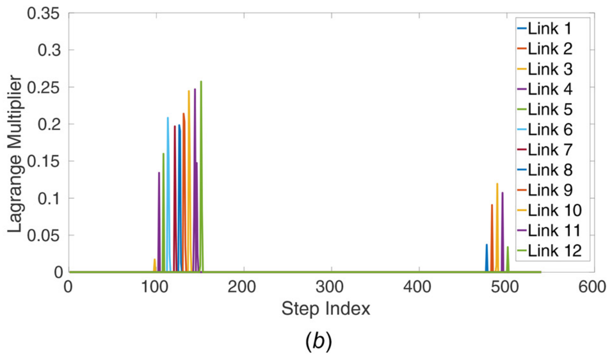


**Fig. 8 Simulation parameter plots: (*a*) plot of head, tail, and intermediate joint angles and (*b*) plot of Lagrange multipliers for first elliptical obstacle for all links**

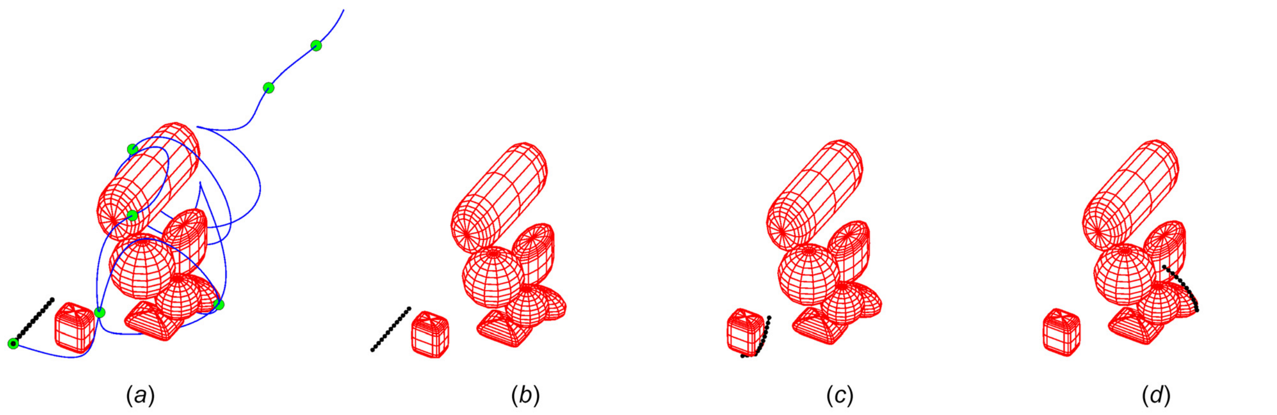
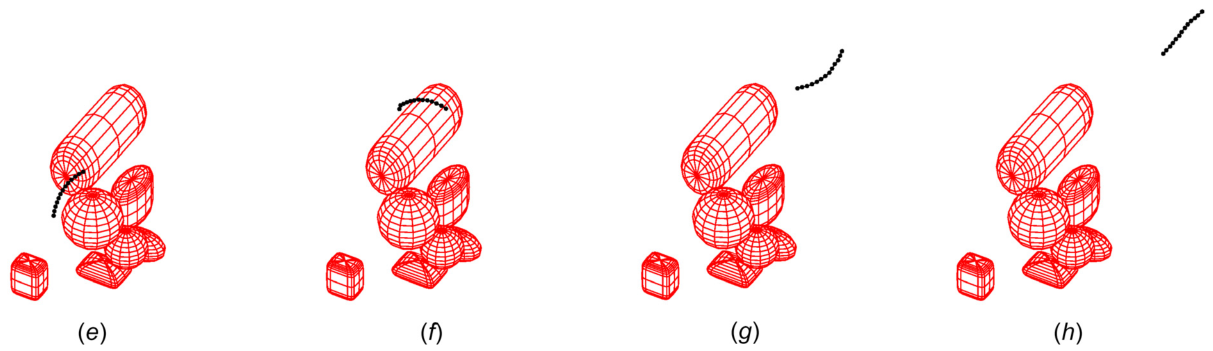



Fig. 9 Motion snapshots for 3D simulation: (*a*) trajectory with snapshot locations and initial configuration, (*b*) snapshot at 1, (*c*) snapshot at 2, (*d*) snapshot at 3, (*e*) snapshot at 4, (*f*) snapshot at 5, (*g*) snapshot at 6, and (*h*) snapshot at 7

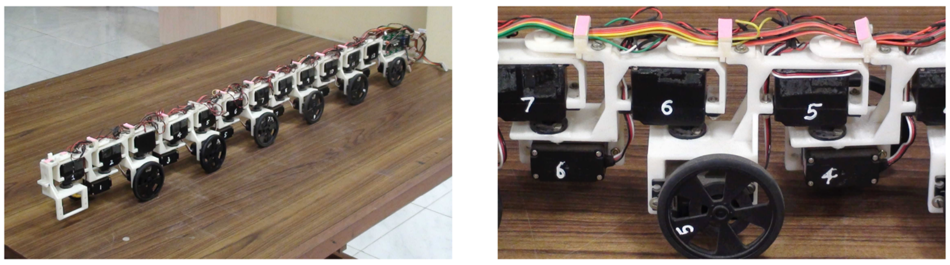
Table 1 Variation of execution time with number of links

| No. | # of links ($n$) | Execution time (s) |
|---|---|---|
| 1 | 6 | 154.87 |
| 2 | 12 | 317.77 |
| 3 | 30 | 715.61 |
| 4 | 50 | 1172.30 |

(middle), and 12 (tail) of the snake robot—over the entire path chosen for the head. It is seen that in the free space, at the start of the path, the motion of the joint angles fall off toward the tail as predicted by the tractrix-based approach. At path points close to the obstacles, all the joint movements are similar, and they are such that the entire body of the snake robot avoids the obstacle.

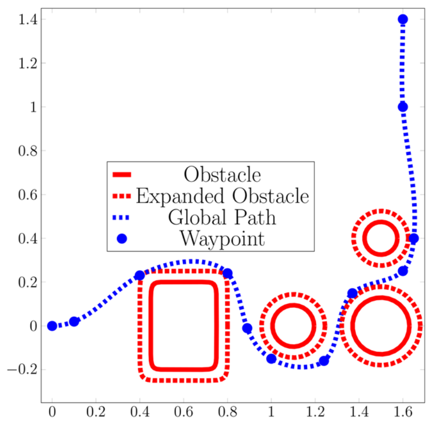
The top half of Fig. 12 shows a sequence of snapshots of the simulated robot configuration along the trajectory, and the bottom half shows the corresponding robot configuration (a video clip of the entire motion is also provided, which is available under the "Supplemental Materials" tab for this paper on the ASME Digital Collection). The path consisted of 12 way-points specified manually for the robot head to pass between the obstacles, with initial and end point orientations specified. The path of the head was calculated by fitting a spline through the 12 way-points, and joint configurations were calculated for 132 steps at a spacing of 25 mm along the spline, using the optimization algorithm with obstacle avoidance constraints. The calculation of the spline through all the way-points took 242 ms, and the computation of the joint configurations with obstacle avoidance for each step of movement took 244 ms on average, with a worst case time of $\sim$2.23 s for the first step and an average time of 228 ms for subsequent steps. The joint angles and wheel speeds were further interpolated in 30 steps, for smoothness of motion, before being fed to the robot. The commands to the robot take approximately 100 ms to be transferred over a slow (9600 bps) serial link. It may be mentioned that computation times can be improved with code optimization and dedicated hardware.

The main objective of the experiments with the fabricated hyper-redundant robot was to verify the feasibility of implementing the tractrix-based algorithm on a physical prototype. It may be noted that the usage of wheels results in nonholonomic constraints



(*a*) (*b*)

Fig. 10 Experimental prototype: (*a*) experimental 12DOF hyper-redundant robot and (*b*) close-up of joint and wheel assembly of the experimental robot



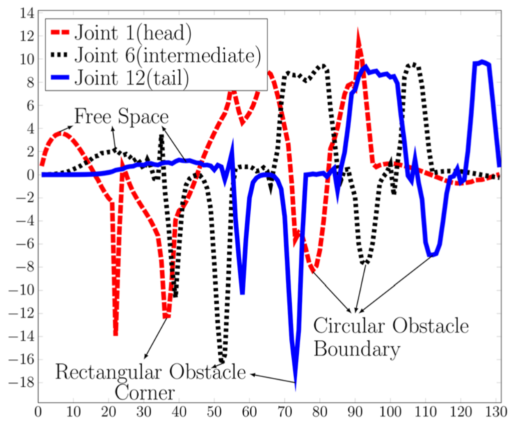


**Fig. 11 Experimental setup and results: (*a*) workspace with obstacles and desired path of the robot head and (*b*) comparison of joint angles at various points in the body**

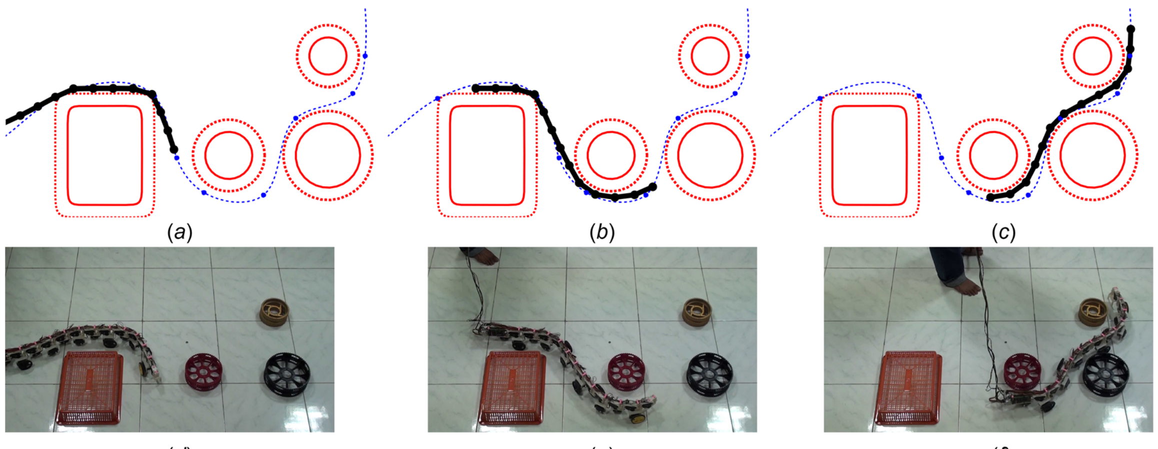


**Fig. 12 Simulated (*a–c*) and actual (*d–e*) configuration of the robot at steps 45, 65, and 97 of the motion: (*a,d*) step 45, (*b,e*) step 65, and (*c,f*) step 97**

on the wheel–ground contact points of the robot, and the wheel slip and other dynamic effects are not taken into account in the tractrix-based algorithm. We have tried to minimize these effects by moving slowly and on a hard flat floor. One consequence of the wheel slip and other unmodeled dynamics is that the path traced is not exactly the same as the desired path. Nevertheless, the hyper-redundant robot using the tractrix-based motion planning algorithm avoids the obstacles, and the path followed by the prototype is reasonably close to the desired path, as seen in Fig. 12.

## 5 Conclusions

In this paper, we have presented an efficient optimization-based approach to motion planning with obstacle avoidance for extended bodies, such as hyper-redundant snakelike robots. Apart from the general theoretical results, numerical simulation results have been presented for motion of a hyper-redundant robot in a plane and 3D space. Simulation results for a 12-link snake robot moving in a field of obstacles have been presented. Experimental results from a prototype 12-link snake robot are seen to be very close to the simulation results and validate the optimization-based algorithm. The algorithm is also amenable to efficient implementation with on-board sensing of the obstacles in real-time. This will form a part of our future work in this space.

In the present setup, the lack of wheel speed and joint position feedback results in a certain amount of wheel slippage at certain points in the trajectory. Extensions planned to this work include an in-depth analysis of the Lagrangian multipliers and also improving the mechanism to reduce slippage in motion.

In this work, the obstacles are represented by smooth, differentiable functions due to the requirement of the gradient-based optimization algorithm used in this work. This is not a serious constraint, and one can have piecewise smooth obstacles and one can also use other optimization algorithms. These are also some of the planned extensions to this work.

## Acknowledgment

We thank Shri K. Ramachandra, Scientist, and Shri P. C. Prakash, Technical Officer, of CAIR for their invaluable help in the design, fabrication, and assembly of the experimental snake robot. This work was partly funded by a grant from the Robert Bosch Centre for Cyber Physical Systems (RBCCPS) at the Indian Institute of Science Bangalore.

## References

[1] Miller, G. S., 2002, "13 Snake Robots for Search and Rescue," *Neurotechnology for Biomimetic Robots*, MIT Press, Cambridge, MA, p. 271.
[2] Sreenivasan, S., Goel, P., and Ghosal, A., 2010, "A Real-Time Algorithm for Simulation of Flexible Objects and Hyper-Redundant Manipulators," Mech. Mach. Theory, **45**(3), pp. 454–466.
[3] Menon, M. S., Ananthasuresh, G. K., and Ghosal, A., 2013, "Natural Motion of One-Dimensional Flexible Objects Using Minimization Approaches," Mech. Mach. Theory, **67**, pp. 64–76.

[4] Hwang, Y. K., and Ahuja, N., 1992, "Gross Motion Planning—A Survey," ACM Comput. Surv., **24**(3), pp. 219–291.
[5] Brooks, R. A., 1983, "Planning Collision-Free Motions for Pick-and-Place Operations," Int. J. Rob. Res., **2**(4), pp. 19–44.
[6] Lozano-Perez, T., 1981, "Automatic Planning of Manipulator Transfer Movements," IEEE Trans. Syst., Man Cybern., **11**(10), pp. 681–698.
[7] Fox, D., Burgard, W., and Thrun, S., 1997, "The Dynamic Window Approach to Collision Avoidance," IEEE Rob. Autom. Mag., **4**(1), pp. 23–33.
[8] Khatib, O., 1986, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," Int. J. Rob. Res., **5**(1), pp. 90–98.
[9] Barraquand, J., Langlois, B., and Latombe, J.-C., 1992, "Numerical Potential Field Techniques for Robot Path Planning," IEEE Trans. Syst., Man Cybern., **22**(2), pp. 224–241.
[10] Rimon, E., and Koditschek, D. E., 1992, "Exact Robot Navigation Using Artificial Potential Functions," IEEE Trans. Rob. Autom., **8**(5), pp. 501–518.
[11] Ó'Dúnlaing, C. P., Sharir, M., and Yap, C. K., 1983, "Retraction: A New Approach to Motion-Planning," 15th Annual ACM Symposium on Theory of Computing, Boston, MA, Apr. 25–27, pp. 207–220.
[12] Glasius, R., Komoda, A., and Gielen, S. C. A. M., 1995, "Neural Network Dynamics for Path Planning and Obstacle Avoidance," Neural Networks, **8**(1), pp. 125–133.
[13] Boyse, J. W., 1979, "Interference Detection Among Solids and Surfaces," Commun. ACM, **22**(1), pp. 3–9.
[14] Prescott, T. J., and Mayhew, J. E., 1991, "Obstacle Avoidance Through Reinforcement Learning," Neural Information Processing Systems (NIPS), Denver, CO, Dec. 2–5, pp. 523–530.
[15] Suh, S.-H., and Shin, K. G., 1988, "A Variational Dynamic Programming Approach to Robot-Path Planning With a Distance-Safety Criterion," IEEE Trans. Rob. Autom., **4**(3), pp. 334–349.
[16] Gilbert, E. G., and Johnson, D. W., 1985, "Distance Functions and Their Application to Robot Path Planning in the Presence of Obstacles," IEEE Trans. Rob. Autom., **1**(1), pp. 21–30.
[17] Luh, J. Y. S., and Lin, C. S., 1981, "Optimum Path Planning for Mechanical Manipulators," ASME J. Dyn. Syst., Meas. Control, **103**(2), pp. 142–151.
[18] Sundar, S., and Shiller, Z., 1997, "Optimal Obstacle Avoidance Based on the Hamilton–Jacobi–Bellman Equation," IEEE Trans. Rob. Autom., **13**(2), pp. 305–310.
[19] Seshadri, C., and Ghosh, A., 1993, "Optimum Path Planning for Robot Manipulators Amid Static and Dynamic Obstacles," IEEE Trans. Syst., Man Cybern., **23**(2), pp. 576–584.
[20] Hanafusa, H., Yoshikawa, T., and Nakamura, Y., 1981, "Analysis and Control of Articulated Robot With Redundancy," 8th Trennial World Congress of IFAC, Kyoto, Japan, Vol. 4, pp. 1927–1932.
[21] Maciejewski, A. A., and Klein, C. A., 1985, "Obstacle Avoidance for Kinematically Redundant Manipulators in Dynamically Varying Environments," Int. J. Rob. Res., **4**(3), pp. 109–117.
[22] Lozano-Perez, T., 1983, "Spatial Planning: A Configuration Space Approach," IEEE Trans. Comput., **C-32**(2), pp. 108–120.
[23] Branicky, M. S., and Newman, W. S., 1990, "Rapid Computation of Configuration Space Obstacles," IEEE International Conference on Robotics and Automation (ICRA), Cincinnati, OH, May 13–18, pp. 304–310.
[24] Lozano-Perez, T., 1987, "A Simple Motion-Planning Algorithm for General Robot Manipulators," IEEE Trans. Rob. Autom., **3**(3), pp. 224–238.
[25] Chirikjian, G. S., and Burdick, J. W., 1992, "A Geometric Approach to Hyper-Redundant Manipulator Obstacle Avoidance," ASME J. Mech. Des., **114**(4), pp. 580–585.
[26] Choset, H., and Henning, W., 1999, "A Follow-the-Leader Approach to Serpentine Robot Motion Planning," J. Aerosp. Eng., **12**(2), pp. 65–73.
[27] Zhang, Y., and Wang, J., 2004, "Obstacle Avoidance for Kinematically Redundant Manipulators Using a Dual Neural Network," IEEE Trans. Syst., Man, Cybern., Part B: Cybern., **34**(1), pp. 752–759.
[28] Kahn, M. E., and Roth, B., 1971, "The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains," ASME J. Dyn. Syst., Meas. Control, **93**(3), pp. 164–172.
[29] Transeth, A. A., Leine, R. I., Glocker, C., Pettersen, K. Y., and Liljebäck, P., 2008, "Snake Robot Obstacle-Aided Locomotion: Modeling, Simulations, and Experiments," IEEE Trans. Rob., **24**(1), pp. 88–104.
[30] Liljebäck, P., Pettersen, K. Y., Stavdahl, Ø., and Gravdahl, J. T., 2014, "A 3D Motion Planning Framework for Snake Robots," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, IL, Sept. 14–18, pp. 1100–1107.
[31] Pfotzer, L., Staehler, M., Hermann, A., Roennau, A., and Dillmann, R., 2015, "KAIRO 3: Moving Over Stairs & Unknown Obstacles With Reconfigurable Snake-Like Robots," European Conference on Mobile Robots (ECMR), Lincoln, UK, Sept. 2–4, pp. 1–6.
[32] Vonásek, V., Saska, M., Winkler, L., and Přeučil, L., 2015, "High-Level Motion Planning for CPG-Driven Modular Robots," Rob. Auton. Syst., **68**(1), pp. 116–128.
[33] Reznik, D., and Lumelsky, V., 1994, "Sensor-Based Motion Planning in Three Dimensions for a Highly Redundant Snake Robot," Adv. Rob., **9**(3), pp. 255–280.
[34] Weinstock, R., 1952, *Calculus of Variations: With Applications to Physics and Engineering*, McGraw-Hill, New York.
[35] Smith, D., 1998, *Variational Methods in Optimization*, Dover Publications, Mineola, NY.
[36] Steinhaus, H., 1969, *Mathematical Snapshots*, Oxford University Press, New York.
[37] Spanier, E. H., 1994, *Algebraic Topology*, Vol. 55, Springer Science & Business Media, New York.
[38] Gardner, M., 1977, *Mathematical Carnival: A New Round-Up of Tantalizers and Puzzles From Scientific American*, Vintage Books, New York.
[39] Barr, A. H., 1981, "Superquadrics and Angle-Preserving Transformations," IEEE Comput. Graphics Appl., **1**(1), pp. 11–23.
[40] MATLAB, 2012, "MATLAB, Version 7.12.0 (R2011a)," The MathWorks, Inc., Natick, MA.