# Motion planning of hyper-redundant robot through confined paths

K. P. Ashwin*, Arkadeep N. C. *, and A. Ghosal †

**Abstract**

Application of highly articulated hyper-redundant robots to manoeuvre in narrow and confined spaces is gaining popularity due to their obvious advantages. Given a path which the end-effector has to traverse, finding the configuration which avoids collision with the environment constitutes obstacle-avoided redundancy resolution and subsequently, the motion planning of the robot. While there are many redundancy resolution methods which address obstacle avoidance, tractrix based optimization method produces displacement of links which attenuates from the end-effector tip to the base of the robot and thereby simulating a more 'natural' motion. While tractrix based methods are studied for obstacle avoidance, application of the method for robots confined within a narrow bounded path(duct) is not very well studied. This paper presents different ways in which ducts can be modelled in 2D plane as well as in 3D space and how motion planning can be achieved using tractrix based methods in each cases. Fundamental formulations and well as more efficient formulations for practical implementation have been discussed. The concepts are demonstrated using simulations conducted on three practical scenarios: 1) hyper-redundant manipulators inspecting an industrial pipeline, 2) motion of an endoscopic robot through GI tract and 3) motion of hyper-redundant manipulators in search and rescue operations. Analysis on the computational complexity shows that the method is feasible for practical implementation.

**Keywords**:6 keywords

## 1 Introduction

In the reachable workspace of a serial robot in 3D, since a given position and orientation of end-effector can be achieved using a maximum of six joints, additional joints would create

*Graduate Student at the Robotics and Design Lab, Department of Mechanical Engineering, Indian Institute of Science, Bangalore 560012, India, email: ashwinkp@iisc.ac.in

†Corresponding Author, Professor, Department of Mechanical Engineering, Indian Institute of Science, Bangalore, email: asitava@iisc.ac.in.

a redundant system. Such robots are generally termed redundant serial robots. While robots with 6 joints or less can achieve a position and orientation in space with limited configurations, there are theoretically infinite possibilities for a hyper-redundant robot to achieve the same. These additional degrees of freedom can be effectively used in situations where the robot has to selectively choose configurations without altering the end-effector pose, such as robotic manipulation through confined spaces. This renders the potential use of highly articulated hyper-redundant robots in applications such as exploring earthquake hit regions for search and rescue operations [1], [2], remotely operated robotic arms for minimally invasive laparoscopic and endoscopic surgeries [3], [4], [5], [6] and inspection of industrial pipelines and nuclear reactors [8], [7], [9], [10], [11]. A large number of redundant joints also make the robots analogous in morphology to snake, elephant trunk, tentacle etc. and hence, a topic of particular interest in bio-inspired robotics [14], [13], [12]. Identifying a suitable configuration from the many possible configurations, given the position and orientation of the end-effector constitutes the redundancy resolution of hyper-redundant robots. If the path through which the end-effector has to follow is predetermined, then the redundancy resolution problem also constitutes the motion planning of the robot.

The initial focus of researchers on addressing redundancy resolution problem was to solve the equation $\dot{\vartheta} = J(\varphi)\dot{\varphi}$ where $J$ is the manipulator Jacobian matrix and $\vartheta$ and $\varphi$ are the vectors in task space and joint space respectively. Primary idea of inverse kinematics is to invert the Jacobian using Moore-Penrose inverse(pseudo inverse). While this method results in kinematic singularities, methods such as extended Jacobian [15], [16] and task prioritized augmented Jacobian [17] are aimed to avoid these singularities and also obtain solutions to inverse kinematic problem by specifying a velocity component at the singular configuration depending on the task at hand. Since the major advantage of using a hyper-redundant robot is to facilitate motion avoiding obstacles and movement in confined spaces, many researchers have worked on this particular task using above mentioned techniques [18]. However, even though the methods avoid singularities in the manipulator Jacobian, it could generate other forms of singularities and also, the method becomes computationally complex for large number of joints [19]. Many other redundancy resolution methods including variational approach, geometric approach, neural networks and fuzzy logic can also be found in the literature [20], [21], [23], [22]. In [8], obstacle avoidance problem for hyper-redundant manipulator is carried out by fitting a curve through the joints of manipulator and planning the path for this 'backbone curve' which avoids obstacles. Finding the pose of the backbone curve directly gives the co-ordinates of the joints. Hence, in case of obstacle cluttered environments, planning the path of the end-effector which avoids the obstacles will

result in redundancy resolution since the trajectory itself forms the backbone curve of the manipulator [25], [24], [26], [?].

Even though the backbone curve approach is very simple, end-effector trajectory with many kinks may sometimes produce undesirably high acceleration at the tail of the link as shown in figure 1. The motion will also look un-natural. In [27], the authors proposed a tractrix based redundancy resolving algorithm which produces natural looking motion of hyper-redundant robot. It is also shown in [28] that the same can be achieved by minimizing the subsequent motion of the links, given the displacement of the head(end-effector), and the motion attenuates from the head to the tail(base) of the entire manipulator. For obstacles represented by simple analytical shapes, the algorithm can be effectively used for obstacle avoidance of the entire link chain as well [29]. However, in the special case of motion through confined spaces within a narrow bounded path (which hereafter we call "duct"), directly implementing the algorithm would require modeling the entire half space outside the duct as obstacles which is impractical. In this paper, we discuss a few methods to represent ducts and how to implement tractrix-based algorithm for motion planning of a hyper-redundant robot. In section 2, an overview of tractrix based algorithm and the physical tractability of the algorithm is discussed. Representation of ducts in 2D plane and motion planning through the planar ducts is detailed in section 3. Representation of ducts in 3D and motion planning in the same is explained in section 4. In section 5, we demonstrate the concepts by simulating the motion of hyper-redundant robots in three practical applications. Advantage of this method in terms of computational complexity and the limitations of this method is also mentioned in the section. Finally, in section 6, we present the conclusions of the paper and the scope of future work.

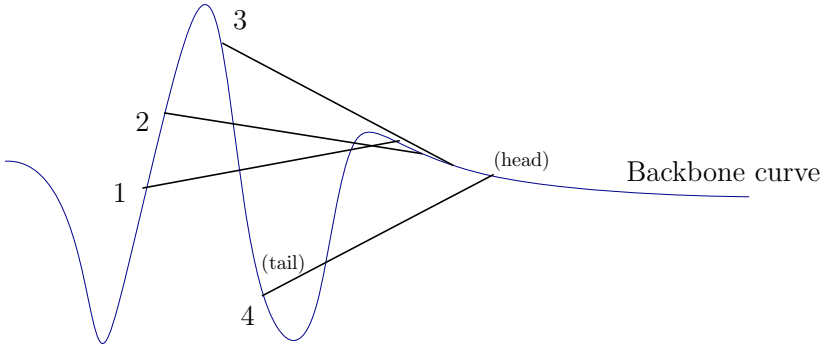



Figure 1: Large motion in tail corresponding to small head movement for single link

## 2 Overview of tractrix based motion planning

Consider a rigid link of length $L_0$ positioned in a 2D plane, initially aligned to the Y-axis as shown in figure 2a . The co-ordinates of the 'head' of the link is given as $\mathbf{X}_h = [X_h, Y_h]^T$ and the co-ordinates of the 'tail' as $\mathbf{X}_t = [X_t, Y_t]^T$. If the head is displaced to the co-ordinate $\mathbf{x}_h = [x_h, y_h]$ along the positive $X-$ axis by $t$ units, the tail of the link can lie anywhere on the circumference of a circle centered at the co-ordinate $(t, 0)$ with radius $L_0$. If we assign a rule that the velocity of the tail of the link is always directed towards the length of the link, we get two diametrically opposite points on the circle. The continuous path traced by the tail point is the well known tractrix curve as given in equation (1):

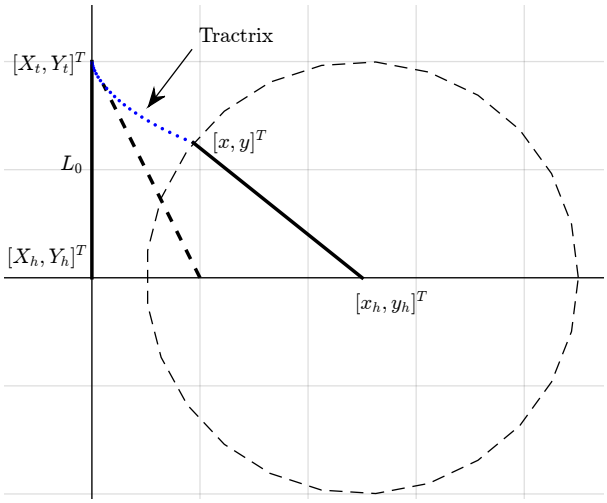
$$\mathbf{x}_t = [x(t), y(t)] = [t - L_0 \tanh \frac{t}{L_0}, L_0 \operatorname{sech} \frac{t}{L_0}] \tag{1}$$

The extension of tractrix equation along motion in arbitrary direction as well as an algorithm to calculate the same in 3-D can be found in [27]. In case of multiple links connected to each other as in the case of a hyper-redundant robot, or a one dimensional object approximated as a series of connected linkages, the algorithm can be applied iteratively from the head to tail as shown in their paper. By moving along the tractix curve, the tail moves the minimum distance with respect to its initial position. Also, the displacement $\|\mathbf{x}_h - \mathbf{X}_h\| \geq \|\mathbf{x}_t - \mathbf{X}_t\|$ which means that the displacement attentuates from the displaced link to end of the chain in case of serially connected links as shown in figure 2b. Due to the minimal displacement of the tail, the motion can be imagined as the one with high lateral resistance on the link and less resistance in the direction of motion of link.

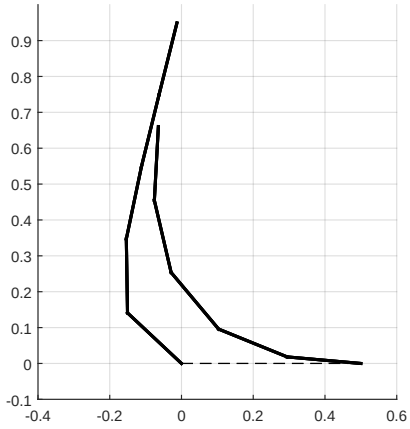
It is shown in [28] that for a single rigid link, tractrix curve can also be obtained by minimizing an $L^2$ metric – the displacement of the tail from its initial position subject to the condition that the length of link is always preserved. So, the co-ordinates of the tail can be obtained from the following minimization problem:

$$\begin{gathered} \arg\min_{\mathbf{x}_t} \|\mathbf{x}_t - \mathbf{X}_t\| \\ \text{Subject to: } \|\mathbf{x}_h - \mathbf{x}_t\| - L_0 = 0 \end{gathered} \tag{2}$$

An advantage of expressing tractrix as a minimization problem is that we can add more constraints to the above expression and hence, control the position of the tail point. Though the resulting curve may not necessarily be a tractix curve, the motion of the tail will appear realistic. For the motion of a link which minimizes its tail velocity (displacement of the tail




(a) Tractrix curve in 2D with one link




(b) Tractrix with multiple links

Figure 2: Tractrix in a plane

co-ordinates), obstacle avoidance is achieved by formulating the problem as:

$$\begin{aligned} &\arg\min_{\mathbf{x}_t} \|\mathbf{x}_t - \mathbf{X}_t\| \\ &\text{Subject to:} \|\mathbf{x}_h - \mathbf{x}_t\| - L_0 = 0 \\ &f(\mathbf{x}) \succeq 0 \end{aligned} \tag{3}$$

where $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ are the analytical equations of the boundaries of the surfaces which are to be avoided. For example, if the tail is to avoid a single obstacle represented by a circle with center $(x_c, y_c)$, the expression $f(x) = (x - x_c)^2 + (y - y_c)^2 - r^2 > 0$ ensures that the point $x$ always lies outside the circle of radius $r$. Complex objects can be modelled as a combination of super-ellipses as shown in [29]. In this case, $\mathbf{f}(x)$ will be a vector of all boundary equations $\mathbf{f}(x) = [f_1(x), f_2(x), ...f_m(x)]^T$. It is also worth noting that the value of constraint function in equation (3) will increase or decrease as the point is farther from the curve $\mathbf{f}(x)$; the value being zero on the curve. Hence, this approach can also be imagined as a geometric potential field, with zero potential only at the surface of the obstacle.

The problem of planning the motion of the robot through a duct may be specified as:

$$\begin{aligned} &\arg\min_{\mathbf{x}_t} \|\mathbf{x}_t - \mathbf{X}_t\| \\ &\text{Subject to:} \|\mathbf{x}_h - \mathbf{x}_t\| - L_0 = 0 \\ &C_{\text{ineq}} : \ f(x) < 0 \end{aligned} \tag{4}$$

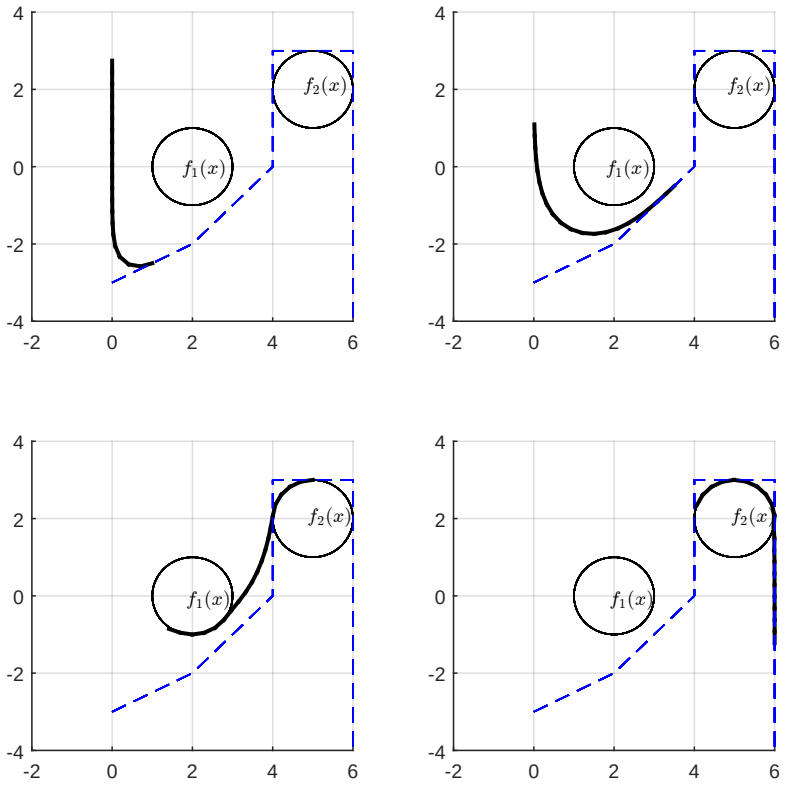



Figure 3: Obstacle avoidance in a plane

While this expression is applicable for a duct represented by a single surface with the boundary $f(x)$, unlike the obstacle avoidance problem, the same will not work in the case of complex surfaces represented by combination of simpler analytical shapes. This is because if a point is classified as inside one of the simpler shapes, then it should be classified as outside the other shapes forming the duct. In other words, if one constraint function $f_k(x) < 0$, then the other constraint functions $f_{i \neq k} \geq 0$. In the next sections, we present different methods to represent the ducts and how confined space motion is achieved in different cases.

## 2.1 Nature of the optimization problem

In the previous section we have discussed how tractrix based motion planning can be formulated as a constrained optimization problem, which entails a discussion on the suitability of the posed optimization problem and guarantee that the solution is unique and physically tractable. To address this, we will prove that the problem in equation (4) and the variations of the same used in the current work can be posed as convex problems. A function

$f : \mathbb{R}^n \to \mathbb{R}$ is *convex* if and only if the following two conditions hold:

a The domain of $f$, $\mathbf{dom}f$ is a convex set, and,

b For all $x, y \in \mathbf{dom}f$ and $\theta$ with $0 \leq \theta \leq 1$

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y) \tag{5}$$

In our problem, the function is the $L_2$ norm of a vector in $\mathbb{R}^2$ or $\mathbb{R}^3$. Though the $L_2$ norm is defined for all real vectors, the constraints ensuring that the object moves within the confinement of the duct restricts the domain to a feasible closed set $\mathcal{S}$[1]. Therefore, if the conditions a and b associated with the definition of a convex function are satisfied for $\mathcal{S}$ and the $L_2$ norm, then the solution obtained for equation (2) is unique.

It is a well known result that any p-norm $||x||_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}$, $x \in \mathbb{R}$ is a convex function for $p \geq 1$. The fact that a p-norm is convex can be shown by using the scalability and the triangle inequality associated with a normed vector-space. With $\theta$ being a scaling parameter, equation (5) simplifies exactly to the triangle equality for norms. Also, any p-norm is a valid norm on $\mathbb{R}$ for $p \geq 1$ follows from the Minkowski inequality[2]. This leaves us to show that the feasible set $\mathcal{S}$ is convex. For all our modeling examples, we choose $\mathcal{S}$ to be convex by assigning convex boundaries to it or, expressing $\mathcal{S}$ as intersections of closed convex sets. However, as is, none of the problems posed in equations (2) to (4) are convex because of the non-linear equality constraints associated with them, which guarantees that the length of a segment is constant. To pose it as a convex problem we approximate the constraint with linear constraints as described in section 5.2. For the cases of motion planning in 2-D and 3-D, the paths chosen are either entirely convex or are discretized as such. As discussed in the following sections, we do not represent a "duct" as a union of closed convex sets as the first condition of convexity cannot be guaranteed for sets formed by union of convex sets.

# 3 Motion planning through planar ducts

In this section, we analyze different methods to represent a duct in 2D planar surface and how we can add constraints in different representations of duct so as to ensure that the the tip will always lie inside the duct during motion. Each method is shown to have its own advantages.

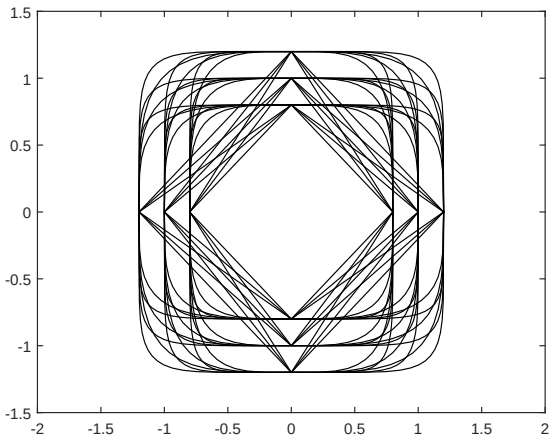
[1] Assuming $\mathcal{S}$ to be a closed set allows the object to physically touch the boundaries of the duct

[2] For a measure space $\mathbb{S}$, $p \geq 1$ and $f$, $g$ members of the Lebesgue space $L_P(\mathbb{S})$, the following inequality holds: $||f+g||_p \leq ||f||_p + ||g||_p$

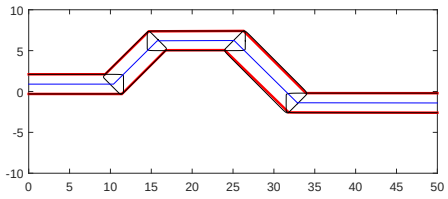
### 3.1 Representation of duct using super-ellipses

One method to represent a duct is by overlapping a series of super-ellipses as shown in 4b. This is the most simple and straightforward means of representation as shown in [29]. In Cartesian co-ordinate system $\mathbb{R}^2$, the contour of super-ellipse obeys the following equation:

$$f(\mathbf{x}) = f(x, y): \quad \left|\frac{x - x_c}{a}\right|^n + \left|\frac{y - y_c}{b}\right|^n - 1 = 0 \tag{6}$$




(a) Super ellipses




(b) Duct modelled as a combination of super-ellipses

Figure 4: Super-ellipses and a duct modelled as combination of super-ellipses

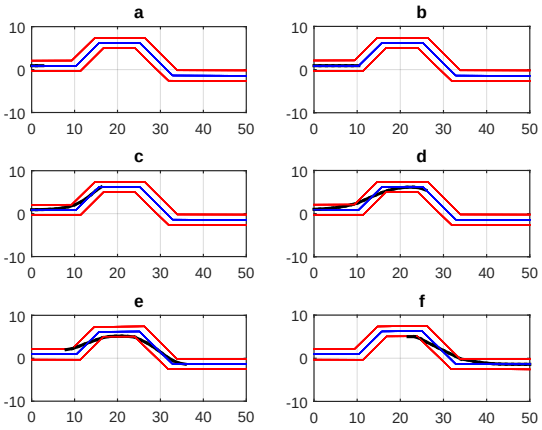
The condition $f(\mathbf{x}_t) < 0$ will ensure that the co-ordinates of the tail of the link $(x_t, y_t)$ always lie inside the bounding curve of a super-ellipse. However, in case of multiple equations $(f_i(x),\ i = 1, 2, ..., m)$, only one of them will be satisfied for the point to be inside the duct. In practical implementation, we can say that this translates to saying that the least value amongst all the values of $f_i(x)$ should be less than zero. For the $i^{\text{th}}$ super-ellipse which is rotated by an angle $\phi_i$ about the $z-$ axis and whose center is translated to the co-ordinates $(x_i, y_i)$ so as to fit a portion of a duct, the co-ordinates of boundary should be multiplied with a transformation matrix

$$T_i = \begin{bmatrix} \cos\phi_i & -\sin\phi_i & 0 & x_i \\ \sin\phi_i & \cos\phi_i & 0 & y_i \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

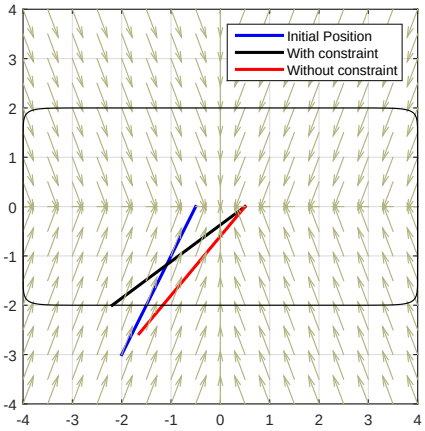
The constraint equation now becomes $g_i(\mathbf{x}_t) : f_i(T_i^T \mathbf{x}_t) < 0,\ i = 1, 2, ..., m$. For practical implementation, we can write the inequality constraint as

$$C_{\text{ineq}}: \quad \min\left(g_i(\mathbf{x}_t)\right) < 0, \quad i = 1, 2, ..., m \tag{8}$$

An example of single link and multi-segmented chain passing through the duct is shown in figure 5a. Motion of a unit link with and without constraint is shown in figure 5b. The negative gradient of the inequality constraint function is also shown in the figure 5b. The method shown here is quite fast and scalable as explained in section section 5.2, while the majority of time taken for the scheme being in identifying the super-ellipsoids which fit the duct profile. For the example shown in this section, this identification is done by manually selecting clusters of points in the duct and fitting ellipses which will reduce the fitting error in a least squared sense.




(a) Motion through duct modelled as combination of super-ellipses [movie link]




(b) Effect of gradient of inequality constraint in pulling the tail into the duct

Figure 5: Tractrix based algorithm on duct represented by ellipsoids

## 3.2 Representation of duct as a set of connected quadrilaterals

Since the profile of a super-ellipse is always symmetric, for complex and non-symmetric ducts, representation using the previous method might require a large number of shapes. For such cases, a complex duct shape can be represented as a closed shape obtained by stitching convex quadrilaterals as shown in 6. The individual quadrilateral patches may be denoted as $A_1, A_2, ..., A_n$, each bounded by the line segments defined by the points $(\mathbf{P_0}, \mathbf{P_1}), (\mathbf{P_1}, \mathbf{P_2}), ..., (\mathbf{P_{n-1}}, \mathbf{P_n})$ for the curve $\zeta_\mathbf{1}$ and $(\mathbf{Q_0}, \mathbf{Q_1}), (\mathbf{Q_1}, \mathbf{Q_2}), ..., (\mathbf{Q_{n-1}}, \mathbf{Q_n})$ for the curve $\zeta_\mathbf{2}$. Classification of a point $\mathbf{x}_t$ as inside or outside a quadrilateral represented by points, say, $\mathbf{P}_1, \mathbf{P}_2, \mathbf{Q}_2$ and $\mathbf{Q}_1$, is essentially checking the placement of the point in the half spaces represented by the four lines spanned by the point set $(\mathbf{P}_1, \mathbf{P}_2)$, $(\mathbf{P}_2, \mathbf{Q}_2)$, $(\mathbf{Q}_2, \mathbf{Q}_1)$,

and $(\mathbf{Q}_1, \mathbf{P}_1)$. This can be written as a set of four inequality constraints:

$$A_i^1 x_t + A_i^2 y_t + B_i < 0 \quad i = 1, 2, 3, 4 \tag{9}$$

$$\tag{10}$$

where $A_i^1 x + A_i^2 y + B_i = 0$ represents a line obtained from one pair of non-diagonal points in the quadrilateral. In matrix form, the inequality will look like:

$$C_{\text{ineq}} : \quad [\mathbf{A}]\, \mathbf{x}_t + \mathbf{B} < 0 \tag{11}$$

where $[\mathbf{A}]$ is a $4 \times 4$ matrix and $\mathbf{B}$ is a $4 \times 1$ vector. But we propose a more convenient method for practical applications which is as follows:

The co-ordinates of a point inside the surface patch $A_i$ is given by the parametric expression

$$\mathbf{x}_i(u, v) = [\mathbf{P}_{i-1} + (1-u)\, \mathbf{P}_i]\, (1-v) + [\mathbf{Q}_{i-1} + (1-u)\, \mathbf{Q}_i]\, (v) \tag{12}$$

in parameters $u$ and $v$. If the vertices of the quadrilateral are given by $\mathbf{P}_i = [{}^x P_i, {}^y P_i]^T$ and $\mathbf{Q}_i = [{}^x Q_i, {}^y Q_i]^T$, then the analytical expressions for the terms $u$ and $v$, given the value of $\mathbf{x}_i$, can be obtained by solving 12 (see Appendix). The values of $u, v$ can be used to classify the point with respect to the surface patch $A_i$[3].

In case of a single quadrilateral patch, the inequality constraints will be simply,

$$C_{\text{ineq}} : \; 0 \leq u \leq 1, \quad 0 < v < 1 \tag{13}$$

for real values of $u$ and $v$. In case of multiple patches, classifying one point with respect to all the patches return the values $(u_1, v_1), (u_2, v_2), ..., (u_m, v_m)$ etc. for the $m$ number of patches $A_1, A_2, ..., A_m$ and consequently, $m$ set of conditions. But out of the $m$ condition sets, only one set should be satisfied since the point will belong to only one patch at a given instance of motion through the duct. Including a switching statement in the optimization code will be inefficient especially when the point to be classified is close to the common boundary between two patches.

In order to provide a gradient to the constraint which will direct the point into the duct, an inequality constraint is to be included as in the case of super-ellipses described in the

[3]It may be noted that there will be two sets of solution and they are not always real and unique. For example, the point $\mathbf{P} = (10, -5)$ when classified with respect to the area $A$ given by the points $\mathbf{P}_1 = (0, 15), \mathbf{P}_2 = (10, 10), \mathbf{Q}_1 = (0, 0)$ and $\mathbf{Q}_2 = (4, 1)$, returns the values $u = (1.0 + 0.6\, i,\ 1.0 - 0.6\, i)$ and $v = (2.0 + 1.9\, i,\ 2.0 - 1.9\, i)$. However, it is easy to filter out the imaginary set of solutions, should the algorithm encounter the same.

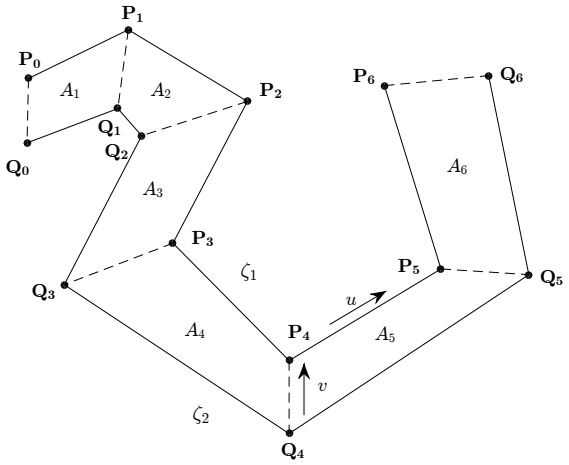



Figure 6: Duct represented by stitched quadrilaterals

previous subsection. If $\hat{u}$ and $\hat{v}$ represent the parameters obtained for a point $\mathbf{x}_t$ classified with respect to the quadrilateral $A_i$, $x_{\zeta_1}(t) = \mathbf{P}_{i-1} + (1-\hat{u})\,\mathbf{P}_i$ and $x_{\zeta_2}(t) = \mathbf{Q}_{i-1} + (1-\hat{u})\,\mathbf{Q}_i$ will give the two points on the duct boundary curves corresponding to the parameter $\hat{u}$. Then we can see that the quantity

$$h = \|\mathbf{x}_{\zeta_1} - \mathbf{x}_t\|^2 + \|\mathbf{x}_{\zeta_2} - \mathbf{x}_t\|^2 - \|\mathbf{x}_{\zeta_1} - \mathbf{x}_{\zeta_2}\|^2 \tag{14}$$

will always assign a negative real value for $h$ when point is inside the duct and a positive real value when the point is outside the duct. The value will be zero only at the boundaries. Hence, for an array of quadrilaterals, it is only necessary that the minimum value of the vector $\mathbf{h} = [h_1, h_2, ..., h_m]$ should be negative for classifying the point with respect to the duct, as in the case of previous section. But as we can see, the inequality only takes into account the parameter variation across the boundaries (along the parameter $v$) and not in the direction of $u$. To account for the same, we make use of the function $\chi$ which is necessarily a linear combination of two Heaviside step functions $H(0) - H(1)$, defined as:

$$\chi(t) = \begin{cases} 0, & t < 0 \\ 1, & 0 \leq t \leq 1 \\ 0, & t > 1 \end{cases} \tag{15}$$

The function $\chi$ applied on the quantity $\hat{u}_i$ (which is the value of parameter $u$ classified with respect to quadrilateral $A_i$), will return 0 only if the point satisfies the constraint $0 \leq \hat{u}_i \leq 1$. Now, multiplying this quantity $\chi(\hat{u})$ with $h_i$ will return a non-zero negative value only if the

point is inside the duct. Then the following inequality constraint:

$$C_{\text{ineq}}: \quad [\chi(\hat{\mathbf{u}})]^T \mathbf{h} < 0 \tag{16}$$

where $\hat{\mathbf{u}} = [\hat{u}_1, \hat{u}_2, ..., \hat{u}_m]^T$ becomes a more practical and convenient way to implement the inequality constraint in the optimization problem.

As an example, motion of a unit link passing through the duct is shown in [fig11] and the effect of the added inequality constraint 16 to pull the tail end of the link which is initially positioned outside the duct, is shown in [fig12]. Apart from being more flexible, another advantage of representing a 2D duct as a set of connected quadrilaterals in this parameteric form is that by setting the limits of the parameter $v$ to $0 + \delta < v < 1 - \delta, \quad \delta < 0.5$, it is easy to manually add a clearance from the walls of the duct without manipulating the duct itself. Also, it is easy to note that $\delta = 0.5 - \epsilon$ (where $\epsilon$ is a small number) would follow the backbone curve motion.

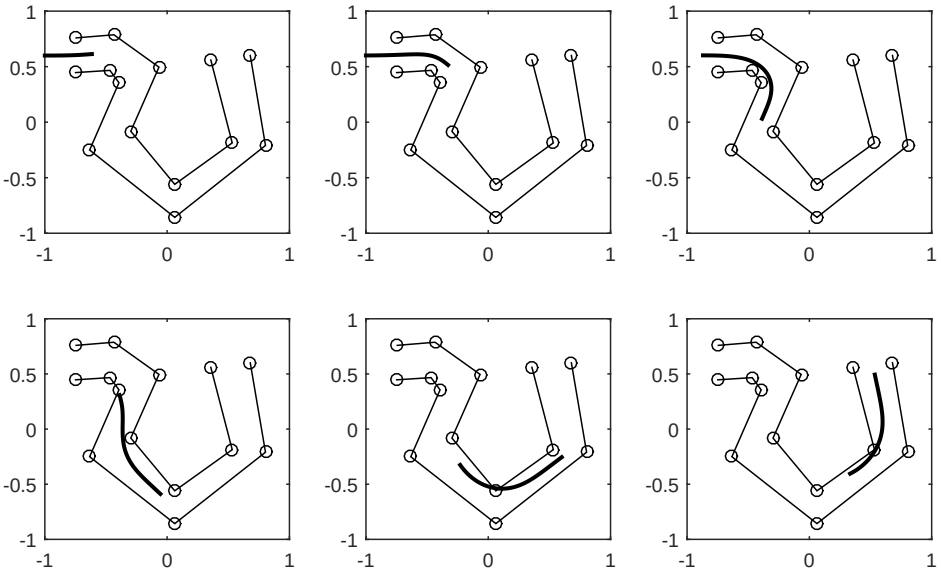


Figure 7: Example of constrained motion with stitched quadrilaterals [movie link]

## 3.3 Representation of duct using two non-intersecting continuous curves

If the non-intersecting border curves of the duct can be analytically expressed, then the equation of the surface patch will simply be,

$$\mathbf{x}_i(u, v) = \zeta_1(u)(1 - v) + \zeta_2(u)(v) \tag{17}$$

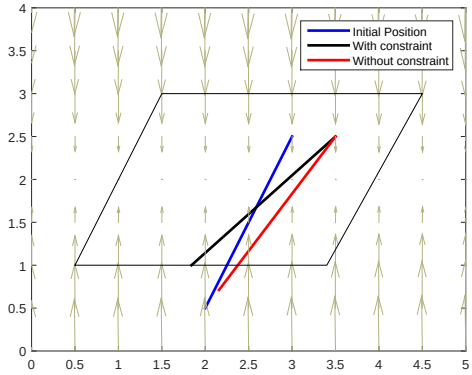
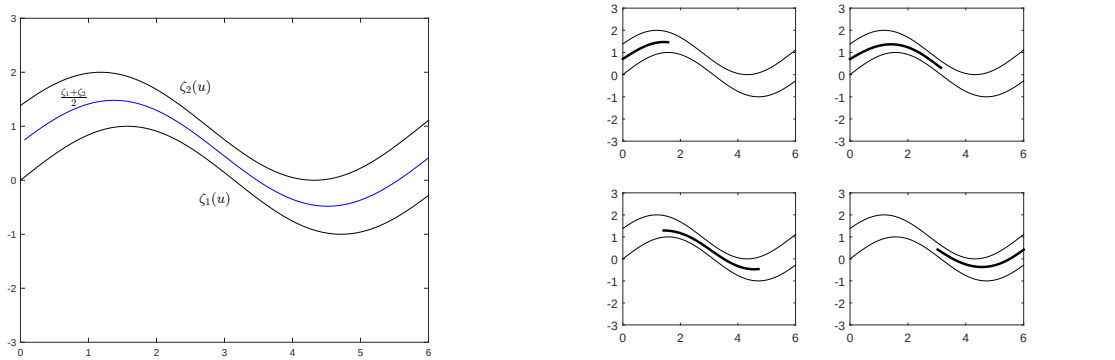



Figure 8: Effect of gradient of inequality constraint in pulling the tail into the quadrilateral duct

For example, [fig11] shows a 2D duct defined by two curves $\zeta_1(u) = [u,\ \sin(u)]^T$ and $\zeta_2(u) = \left[u,\ \sin\left(u + \frac{\pi}{8}\right) + 1\right]^T$ and a path chosen midway between the two curves. The equation of the surface generated by this curves will be




(a) Example of analytical duct

(b) Motion through analytical duct [movie link]

Figure 9: Tractrix based algorithm on analytical duct

$$\begin{bmatrix} x(u,v) \\ y(u,v) \end{bmatrix} = \begin{bmatrix} u \\ \sin(u) + \left[\sin\left(u + \frac{\pi}{8}\right) - \sin(u) + 1\right] v \end{bmatrix} \tag{18}$$

which has the analytical solution for $u$ and $v$:

$$u = x$$
$$v = \frac{y - \sin(x)}{\sin\left(x + \frac{\pi}{8}\right) - \sin(x) + 1}$$

In this case, we will solve the equations:

$$\min_{\mathbf{x}_t} \|\mathbf{x}_t - \mathbf{X}_t\| \tag{19}$$

$$\text{sub:} \quad \|\mathbf{x}_h - \mathbf{x}_t\| - L_0 = 0$$

$$0 < v|_{\mathbf{x}_t} < 1 \tag{20}$$

An example movement of hyper-redundant manipulator through the duct is shown in [fig12]. However, analytical solution is not always viable for complex equations and numerical procedure must be employed to find the values of $u$ and $v$ corresponding to the given tail point to be classified. Also, since multiple solutions may be possible for such cases, the correctness of the solution would heavily depend on the choice of initial guess provided[4]. The equation to be used is 13.

# 4 Motion planning through 3D ducts

In this section, we will explain a few methods to represent ducts in 3D and how motion planning is achieved in the same.

## 4.1 Representation of duct using combination of super-ellipsoids

In Cartesian co-ordinate system in $R^3$, the surface of a super-ellipsoid follows the equation:

$$f(x,y,z) : \left[ \left\{ \left(\frac{x}{a}\right)^{\frac{2}{e}} + \left(\frac{y}{b}\right)^{\frac{2}{e}} \right\}^{\frac{e}{n}} + \left(\frac{z}{c}\right)^{\frac{2}{n}} \right]^{\frac{n}{2}} - 1 = 0 \tag{21}$$

By changing the parameters $a, b, c$ and $n$, we get different closed surfaces as shown in [fig13]. By combining different super-ellipse shapes, we can generate a 3D duct as shown in [fig14]

The procedure to calculate the inside-outside condition is same as that of the method described in section [ref sec]. The inequality condition will be [8]. An example problem with duct approximated using super-ellipsoids is shown in [fig15]. As mentioned in the case for super-ellipses, solution to the motion planning problem with ducts represented by super-ellipsoids is fast (also explained in section ??). Identifying the shapes which fit the duct, is also same as the method mentioned in section ??.

[4]The same argument will also hold for analytical surfaces spanned in 3D and hence is not studied further.

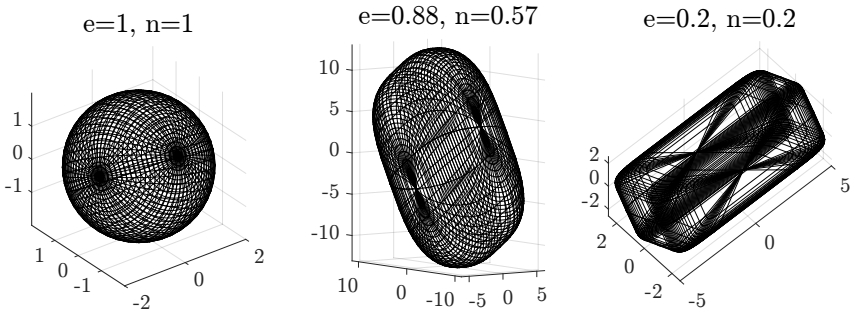



Figure 10: Super-ellipsoids

## 4.2 Representation of duct as a set of connected cylinders

Similar to the set of connected quadrilaterals in 2D, a duct in 3D can be represented by series of connected cylinders. By linearly interpolating two circles in space, we get the parametric equation of the cylinder as (refer Appendix for the detailed expressions):

$$x = C_1(u, t, \theta), \quad y = C_2(u, t, \theta), \quad z = C_3(u, t, \theta) \tag{22}$$

where the parameters $u, t$ and $\theta$ varies along the radial, axial and circumferential direction of the cylinder respectively (refer [fig ??]). Similar to the representation in [sec], $0 \leq u < 1$ and $0 < t < 1$ classifies the point as inside the cylinder. The constrained inequality 16 generated will also be valid for cylinders. The quantity $h_i$ will simply be $h = \hat{u}_i - 1$, which will show the same characteristics as defined by the value of $h_i$ in equation 14. The constraint inequality, hence takes the form:

$$C_{\text{ineq}} : \quad \left[\chi\left(\hat{\mathbf{t}}\right)\right]^T \mathbf{h} < 0 \tag{23}$$

As is the case of quadrilaterals,it is possible to add a clearance from the walls by changing the radius of cylinder from $r$ to $r - \delta$, which is a very desirable characteristic for robots used in medical applications.

## 4.3 Representation of duct as point clouds

The most direct way of representing the duct, in a practical sense, would be as a point cloud ($\mathbf{R}_i, i = 1, 2, ..., N$ where $N$ is the number of points in the cloud) such as obtained from a LiDAR system or a depth map. Subsequently, it would be possible to process the raw data to obtain the geometric representation of the point of cloud as a Stereolithographic

formatted file (STL) or the like. Using the current framework, it is possible to pose the motion planning problem in the following form:

$$\begin{aligned} \min_{\mathbf{x}_t} & \|\mathbf{x}_t - \mathbf{X}_t\| \\ \text{sub:} \quad & \|\mathbf{x}_h - \mathbf{x}_t\| - L_0 = 0 \\ & [\mathbf{A}]\mathbf{x}_t + \mathbf{B} \leq 0 \end{aligned} \tag{24}$$

where $\mathbf{A}$ is a $m \times 3$ matrix and $\mathbf{B}$ is a $m \times 1$ vector. The left hand side of $m$ inequalities represent the equations of $m$ number of planes spanned by three adjacent points in the cloud. The $i^{\text{th}}$ equation, $A_i^1 x_t + A_i^2 y_t + A_i^3 z_t + B_i$ takes a value less than zero when the point $\mathbf{x}_t$ is in the half space which contains the origin and is greater than zero otherwise. The value also provides the attractive gradient which will ensure that the point stays inside the duct. However, in actual implementation, this procedure will be tedious and for practical convenience, it is possible to classify the point $\mathbf{x}_t$ as inside or outside the hull using the algorithm 1 described in section 7.3. The attracting gradient which ensures the point to be inside the duct–as the case with the previous methods– can be provided using the artificial potential field generated from the centroid of the point cloud in conjunction with the output of the in-out function. The inequality constraint then becomes

$$w(\mathbf{R}) \frac{1}{\|(\overline{\mathbf{R}}) - \mathbf{x}_t\|} \leq 0 \tag{25}$$

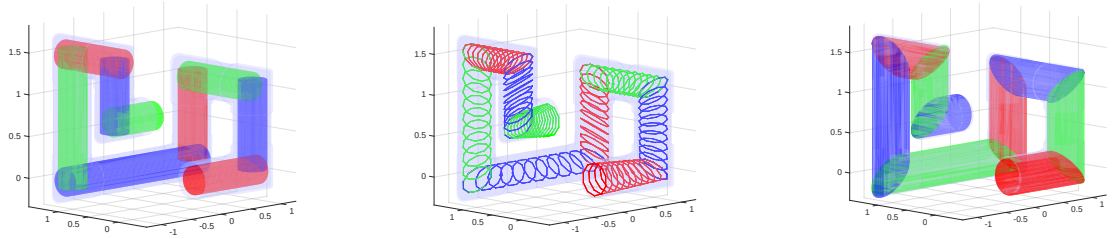
where $w(\overline{\mathbf{R}})$ represents the output from in-out function which is either 1 for the point being outside and 0 for the point being inside the cloud or on the bounding surface[5].

Another interesting prospect of this method is its application in obstacle avoidance where the left hand side of the equation Equation (25) will be greater than zero. The obstacle avoidance methods which follow the equation given by Equation (3) is often limited to shapes which can be modelled analytically. Though the algorithm shown here is limited to convex shapes, they can be applied to more complex and non-symmetric shapes. Representation of pipe using ellipsoids, analytical cylinders and as convex point clouds is shown in 11.

# 5 Examples and discussion

In this section, we present three practical examples where the above mentioned methods are applied and some discussion on the implementation of the above mentioned algorithms.

[5]Unlike the previous classification problems, the bounding surface will also be considered as inside the surface in this case.



(a) Ellipsoid fit (b) Cylinders fit (c) Convex polyhedra

Figure 11: Representation of duct using ellipsoids, cylinders and point clouds

## 5.1 Examples of motion planning through ducts

### 5.1.1 Motion planning for inspection robots

**??** One major use of hyper-redundant robots is in inspection of ducts such as industrial pipelines. By approximating the pipe-line as ellipsoids or cylinders, methods mentioned above are made use of, to plan the motion of a hyper-redundant robot through the same. The path is chosen as the medial axis of the duct, which is also the axis of the cylinders which make up the duct. The configuration of hyper-redundant robot for each path-step is calculated and the simulation of resultant motion is shown in figure 12.

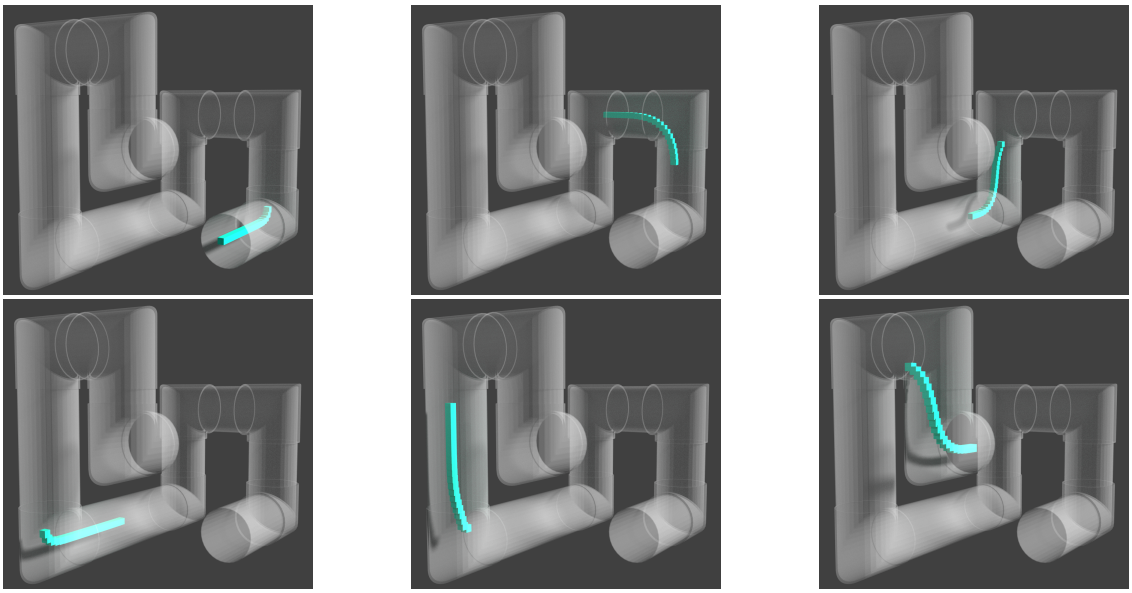


Figure 12: Motion of hyper-redundant manipulator through a duct [movie link]

### 5.1.2 Motion planning through a GI tract

There has been a growing interest in simulating motion of endoscope through GI tract for developing simulators for endoscopy and for implementing path and motion plans for endoscopic and laparoscopic surgical robots. In this section, we simulate the natural motion of an endoscope through GI tract. For simulation, we use the stereolithographic data of GI tract obtained by processing CT scan data. For demonstration, both the methods presented in [sec] and [sec] are used for approximating the GI tract. In the first method, a collection of points are manually selected from the STL file where super-elliposids are fit based on least square error minimization techniques. Representation of GI tract as series of super-ellipsoids is shown in [fig19]

(a) GI tract as collection of super-elliposids

(b) GI tract with connected cylinders

Figure 13: Representation of GI tract in two methods

For representing GI tract as cylinders, we first found out the medial axis of the duct using ??. Then at equal intervals of distance along the medial axis, planes are drawn normal to the same. The collection of points which are in the close proximity of the plane are selected and a circle is fitted on the points using least square error minimization. The parameters so obtained are used for the cylinder equations in [22]. Representation of GI tract as a series of connected cylinders is shown in [fig20]

The realistic motion simulation of endoscope through GI tract is shown in [fig21]

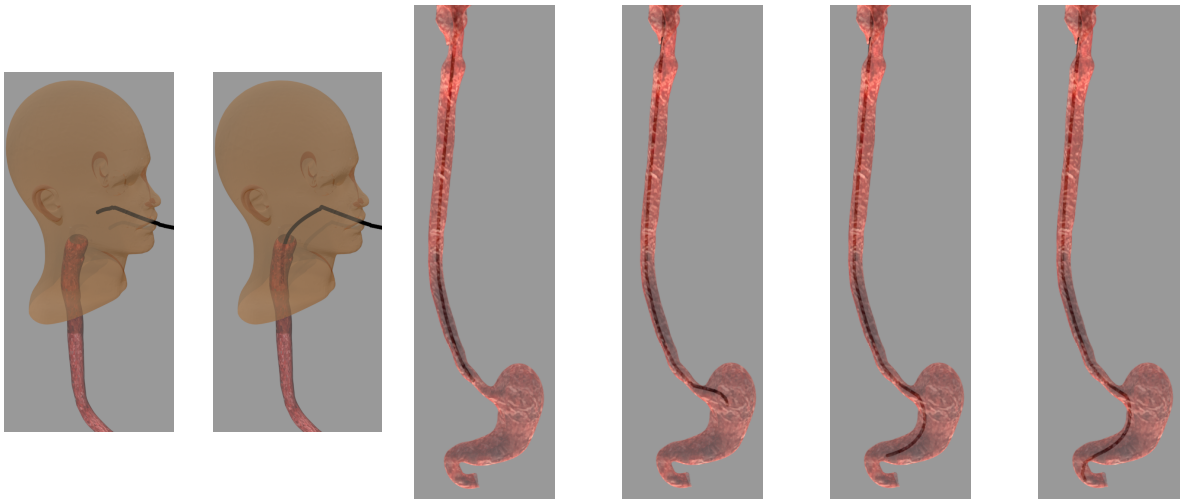


Figure 14: Motion of endoscope through GI tract [movie link]

### 5.1.3 Motion planning for search and rescue operations

Exploring a cluttered environment– such as in an earthquake hit zone is one of the major applications that necessitates the use of hyper-redundant manipulators. In this example, we show how the proposed algorithms can be effectively used in such applications. With reference to figure 15, the objective of the hyper-redundant manipulator, consisting of 20 links is to 1) enter the scene through the hole in the outer wall, 2) Pass through the vent to get inside the room, 3) Explore the objects 3,4 and 5 located inside the blue region and 4) Exit without colliding with the complex shapes given by objects 6 and 7. Here, the exploration problem is effectively tackled by combining the various methods discussed so far. We use the following constraints to define the 6 different problems of the form in equation (4), to generate the entire motion plan. The first problem is to enter through the hole in the wall. For this, an ellipsoid of the size of the hole located in the outer wall, and the feasible set for the first problem is the interior of the ellipsoid, the path being the axis along the direction of the entry. The second problem is a free tractrix motion, where the robot has to reach the opening of the vent. The feasible set in this case is $\mathbb{R}^3$. Next, the robot passes through the vent. This problem is analogous to the one discussed in **??**. A collection of cylindrical polyhedra have been to represent the vent– the feasible set in this problem. This is followed by another free tractrix problem, where the robot exits the vent at the top of the room and reaches to the floor level. Following which, the robot enters the transparent blue search region populated by objects 3 4 and 5. The feasible set of each of the links are obtained on the fly as a combination of the faces of the search region (transparent blue polyhedron)

and the nearest obstacle. Finally, the robot exits the scene through another 'hole' in the designated search region modeled as a cylinder. This guarantees that the complex objects 6 and 7 have no effect on the robot motion as they have no contribution to the feasible set of the final problem. This would have been particularly difficult in half-space based motion plans[ refre midhuns jmr]. Also, it is easy to see that the geometry based global motion planning techniques for obstacle avoidance existing in literature will entail a very involving problem formulation [ananthanarayanan]. Figure 16 shows the simulated results for the motion of manipulator along a chosen path.

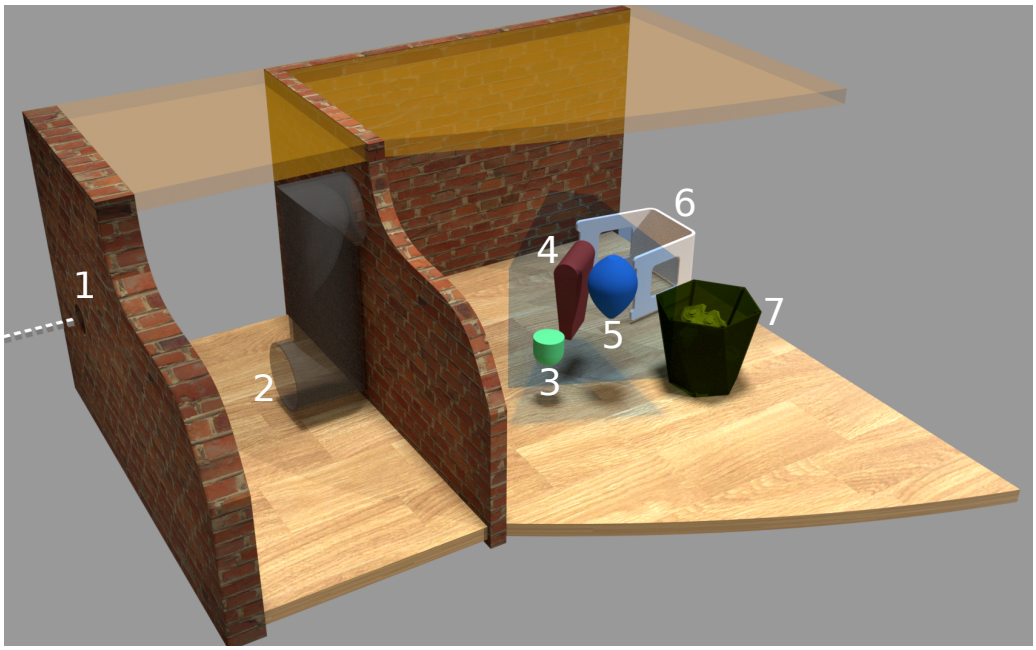



Figure 15: Scene of search and rescue problem

## 5.2 Computational complexity of the proposed scheme

An important aspect of the formulations described in the current work was to show that the formulation and implementation of motion planning problem using tractrix is intuitive and has a broad scope of application. In concurrence to that theme, in this section, we attempt to analyze the worst case computational complexity of such an implementation to show that the current work has potential for real time application in actual problems. We begin by reviewing some mathematical concepts, after Boyd[opt book], to be used in the following discussion.

*Self Concordance*: A convex function $f : \mathbb{R} \to \mathbb{R}$ is *self-concordant* if $|f'''(x)| \leq 2f''(x)^{3/2}$

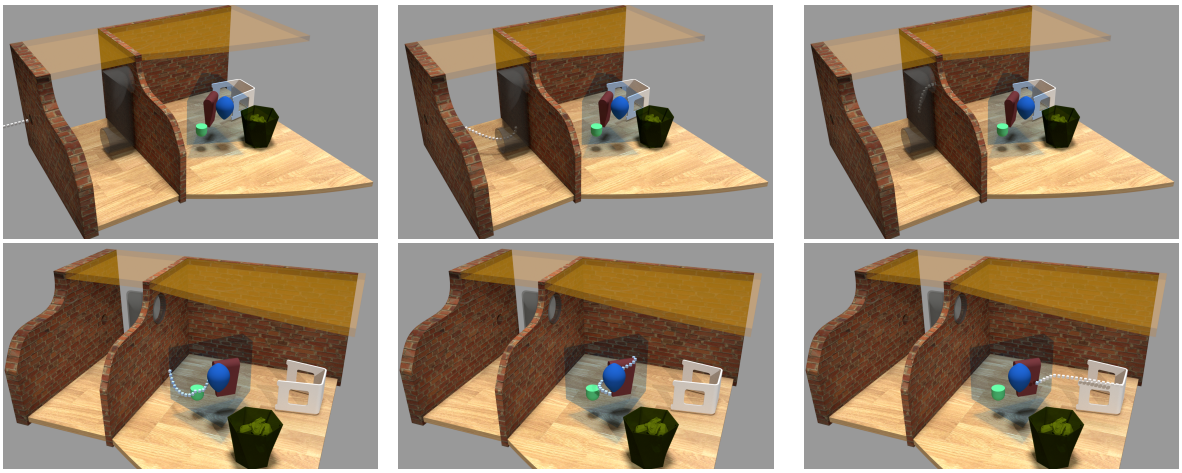


Figure 16: Motion of hyper-redundant robot in confined spaces [movie link]

for all $x \in \mathbf{dom} f$. It can be shown that the Eucledian norm and linear functions are self concordant.

*Feasible set*: For an optimization problem with constraints $f_i(\mathbf{x}) \leq 0$ and $h_j(\mathbf{x}) = 0$, for $i = 1, 2, ..., n$ and $j = 1, 2, ..., m$, the feasible set is $\mathcal{S} = \cap_{i=1}^{n} \mathbf{dom}(f_i \leq 0) \cap \cap_{j=1}^{m} \mathbf{dom}(h_j = 0)$. For all our problems, $\mathcal{S}$ is a subset of the real space $\mathbb{R}$ of dimension 2 or 3.

*Slater's condition*: Slater's conditions of constraint qualification hold when there exists an $\mathbf{x} \in \mathcal{S}$, a strictly feasible point, such that all the inequality conditions and equality conditions are satisfied[6]. This would, in general, mean that strong duality would hold and at the optimum point, the primal and dual problem would have the same solutions.

With these, we move on to our topic of complexity analysis. In section 2.1 we have discussed that the problem as presented in equation (2) is not convex due to the non-linear equality constraint guaranteeing the length of a link is constant at all times. We get around by replacing the non-linear equality constraint by it's affine equivalent– a first order approximation of the constraint about a feasible point $\mathbf{x}^* \in \mathcal{S}$, and a trust region constraint as shown in problem in equation (26). It is imperative for the trust region radius to be small– in our implementation, we have chosen the trust region radius $\epsilon_T$ to be about 6 orders of magnitude smaller than the smallest dimension of $\mathcal{S}$. In equation (26), $\nabla_{\mathbf{x}_t}$ is the gradient

[6]We resort to a slight misuse of notation for brevity. For a more complete treatment, one may refer to Chapter 5 of the book by Boyd and Vandenberghe [Boyd book].

of the objective function.

$$\begin{aligned} \arg\min_{\mathbf{x}_t} \; & g(\mathbf{x}_t) : ||\mathbf{x}_t - \mathbf{X}_t|| \\ \text{Subject to} \; & : g(\mathbf{x}^*) - \nabla_{\mathbf{x}_t} g(\mathbf{x}^*)(\mathbf{x}_t - \mathbf{x}^*) = 0 \\ & |\mathbf{x}^* - \mathbf{x}_t| \preceq \epsilon_T \\ & C_{ineq} : f(x) \preceq 0 \end{aligned} \tag{26}$$

We can rewrite equation (26) as a standard nonlinear program with the objective function $g$, all the $'m'$ constraints $\mathbf{f}$ and $\mathbf{s}$, a vector of $m \times 1$ slack variables as

$$\begin{aligned} \arg\min_{\mathbf{x}_t} \; & g(\mathbf{x}_t) \\ \text{Subject to:} \; & \mathbf{f}(\mathbf{x}_t) + \mathbf{s} = 0 \\ & \mathbf{s} \succeq 0 \end{aligned} \tag{27}$$

We also observe that the problem in equation (26) corresponding to the implementations studied so far, is actually a minimum norm problem with either polyhedral (or polygonal in 2D) or super-quadric (or super-elliptic) constraints. As all of the constraints qualify Slater's condition, we conclude that strong duality holds for all versions of the problem studied in the current work. We choose a interior point method to solve the nonlinear programming problem at hand and by reformulating equation (27) as a barrier problem with barrier parmeter $\mu$. The KKT necessary and sufficient conditions for the given problem is given in equation (28).

$$\begin{aligned} \nabla_{\mathbf{x}_t} g(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \mathbf{f}(\mathbf{x}_t)^T \vec{\lambda} &= 0 \\ \mathbf{f}(\mathbf{x}_t) + \mathbf{s} &= 0 \\ \mathbf{LS} - \mu \vec{e} &= 0 \end{aligned} \tag{28}$$

In equation (28), $\mathbf{L}$ is the $m \times m$ diagonal matrix of the Lagrange multipliers ($\vec{\lambda}$), $\mathbf{S}$ is the $m \times m$ diagonal matrix of the slack variables $\mathbf{s}$ and $\mathbf{e}$ is a $m \times 1$ vector of ones. The above system represents $2m + 1$ equations in $2m + 1$ variables and is known as the KKT system. While using an interior point methods, the computation times depend on two principal factors– the number of Newton steps required to converge and the complexity of each Newton step i.e. total amount of computation required in solving equation (28) to obtain the Newton direction. For our case with a self-concordant log barrier and polyhedral constraints, which are by default self-concordant, we can conclude that the total number of Newton steps are of the order of $\sqrt{m} + c$ where $c$ is a integer. In case of our problems, we

actually observed this phenomenon where, for a 2D case the total number of iterations were about 10, for $m = 6$ for 4 linear constraints defining $\mathcal{S}$ and 2 trust region constraints. Equation (28) can be linearized, and solved (for a 3D case) to obtain the Newton direction as

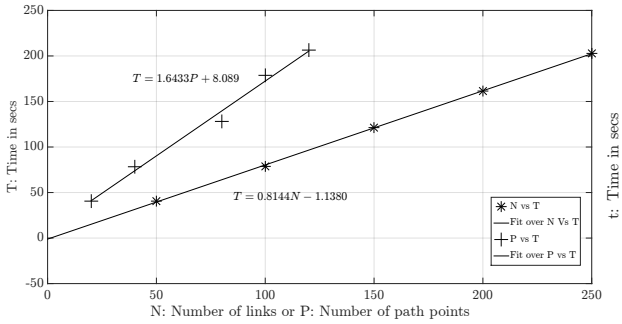
$$\begin{pmatrix} A_{3\times3} & B^T_{m\times3} & 0_{3\times m} \\ B_{3\times m} & 0_{m\times m} & I_{m\times m} \\ 0_{6\times m} & \mathbf{S} & \mathbf{L} \end{pmatrix} \begin{pmatrix} \triangle\, x \\ \triangle\lambda \\ \triangle s \end{pmatrix} = \begin{pmatrix} -\mathbf{F}_1 \\ -\mathbf{F}_2 \\ -\mathbf{F}_3 \end{pmatrix} \tag{29}$$

where, $\mathbf{A} = \nabla^2_{\mathbf{x}_t} g(\mathbf{x}_t) + \sum_{i=1}^{m} \nabla^2_{\mathbf{x}_t} \lambda_i f_i(\mathbf{x}_t)$ is a $3 \times 3$ symmetric matrix, $B = \nabla_{\mathbf{x}_t} \mathbf{f}(\mathbf{x}_t)$ is a $m \times 3$ matrix of the gradient of the $m$ constraint functions, and $\mathbf{F}_1$, $\mathbf{F}_2$, and $\mathbf{F}_3$ are the left hand sides of equation (28). Following [put nilanjan's ieee tro paper] we can obtain the Newton directions as:

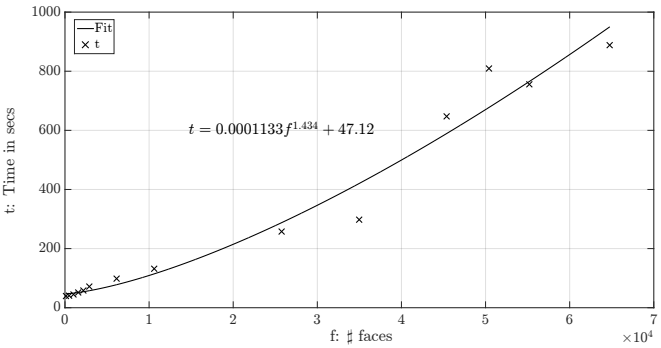
$$\begin{aligned} \triangle\mathbf{x}_t &= \mathbf{G}^{-1}(-\mathbf{F}_1 - \mathbf{B}^T(\mathbf{S}^{-1}\mathbf{L})(\mathbf{F}_2 + \mathbf{L}^{-1}\mathbf{F}_3)) \\ \triangle\lambda &= (\mathbf{S}^{-1}\mathbf{L})(\mathbf{B}\triangle\mathbf{x}_t - \mathbf{F}_2 + \mathbf{L}^{-1}\mathbf{F}_3) \\ \triangle s &= -\mathbf{L}^{-1}(\mathbf{F}_3 + \mathbf{S}\triangle\lambda) \end{aligned} \tag{30}$$

Where, $\mathbf{G} = \mathbf{A} + \mathbf{B}^T(\mathbf{S}^{-1}\mathbf{L})\mathbf{B}$, a $3 \times 3$ matrix (for a 3D case) which can be inverted symbolically, also, $\mathbf{S}$ and $\mathbf{L}$ are diagonal matrices so their inverse can be calculated in linear time ($\mathcal{O}(m)$). Overall, the asymptotic complexity of solving equation (30) is $\mathcal{O}(m^3)$ as the calculation of $G^{-1}$, $\triangle x$ and $\triangle\lambda$ are $\mathcal{O}(m^3)$. This would result in a total asymptotic complexity of the implementation to be $\mathcal{O}(m^{3.5})$. However, the computation time of an iteration doesn't depend on the number of links of the hyper-redundant robot or the step size path, hence the proposed method is linear in these. This is observed from figure 17a. As discussed in section 4.3, in our implementation of the motion planning problem inside a polyhedral domain, we use a separate algorithm to determine whether to use the constraints of the form $[\mathbf{A}]_{m\times3}\mathbf{x}_t + \mathbf{B}_{m\times1} \leq 0$ in equation (24). Algorithm 1 discusses an $\mathcal{O}(m)$ algorithm to do the same in case of a polyhedron with $m$ faces. Therefore, the total asymptotic complexity of our implementation should be about $\mathcal{O}(m^{1.5})$. Figure 17b shows that actual complexity implementation of our algorithm scales as $(m^{1.43})$, with room for further improvement[7]. Furthermore, the formulations involving parametric shapes (super ellipses and super ellipsoids) are even faster because the classification of a point with respect to the domain is available

[7]Computation times in figure 17 was obtained using the 'tic-toc' function in MatlabR2016b running on a workstation with dual 8 core Intel XEON (2.1GHz), 32 GB RAM and Windows 10. The constant term in the fit equation shown in figure 17b signifies the time required to compute the free tractrix problem given by equation (2)




(a) Our implementation is time-linear in number of path points and link numbers




(b) Our implementation scales as $(m^{1.43})$

Figure 17: Complexity of our implementations

as a closed form expression. It may also be noted that this analysis is not applicable to cylinders since they require iterative methods to solve the in-out classifier.

To summarize, three methods to represent 3D ducts are discussed in the previous section which can be chosen based on the computational time and required modelling effort. In the first method of representing ducts by a collection of ellipsoids, modelling time is significant as a separate utility is required to identify the points on surface and to fit the ellipsoids on the data points. However, the solution method is quite fast due to the availability of a closed form function for the in-out classification. The second method of fitting cylinders also involve significant amount of pre-processing to identify cylinders that fit the duct. The implementation time is also high due to the non-analytical nature of the in-out classifier. However, the provision of having a constant clearance from the wall makes it particularly attractive for formulating motion plans for which the thickness of the robot has to be taken into account. Finally, the method of using convex polyhedra to represent ducts requires the least pre-processing time, and is easy to implement with the worst case complexity of $\mathcal{O}(m^{1.5})$.

## 5.3 Limitations of the tractrix based scheme

In spite of the certain advantages regarding formulation and computational aspects of the tractrix based motion planning approach, there are a few limitations which are of importance. The ducts represented by analytical curves and cylinders would require an in-out classifier to demarcate the feasible space of the posed optimization problem. Often, this would involve numerical formulations which are quite computationally involving.

It can be seen that a given hyper-redundant robot with a particular link length will not

be able to negotiate a path with a "very low" curvature. This is shown in figure 18a. In figure 18a, the points 1,2,...,8 denote the coordinates of $\mathbf{x}_h$ across successive iterations. From iteration 6 onwards, the constraints demarcating the feasible space $\mathcal{S}$ and the one guaranteeing a constant link length cannot be simultaneously satisfied unless the tail $\mathbf{x}_t$ (the result of equation (2)), backtracks its own path. Soon after, the optimization problem stops as the link seems to be "locked" at the trough of $\zeta(u)$[8]. In this section we want to quantify the locking effect. To this end, we borrow the concept of "traversability" from literature on wheeled mobile robots (see e.g. Nilanjan's JMD paper). A curve $\zeta(u)$ is traversable by a circle $C_i$, $C_i : \begin{pmatrix} R_i\cos(v) \\ R_i\sin(v) \end{pmatrix}$, $0 \leq v \leq 2\pi$ if $C_i$ can roll over the curve $\zeta(u)$ while maintaining "only one" point of contact at all times. Traversibility for planar curves can be described by the relative curvature $\kappa_R$ of the circle and the curve at the point of contact. For a planar curve and a circle of radius $R_i$, the relative curvature can be given as

$$\kappa_R = \frac{\zeta_{uu}}{(1+\zeta_u^2)^{3/2}} - \frac{1}{R_i}$$

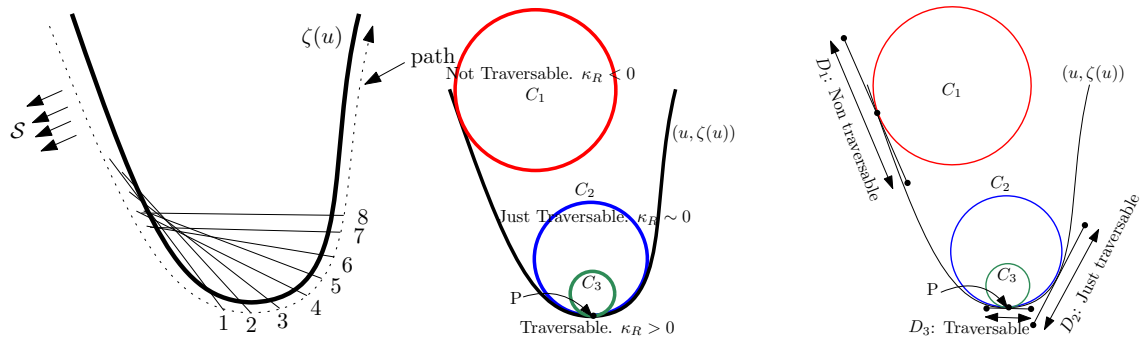
A curve is traversable if the relative curvature is greater than 0, just traversable if it is equal to zero and not traversable if it is negative. This is explained in figure 18b. This concept would generalize to traversable surfaces when the minimum of the two eigenvalues of the relative curvature matrix[9] is positive for traversability or vice versa. Based on this result, we hypothesize that if curve is traversable by a circle of diameter $D_i$, then a hyper-redundant robot of link length $D_i$ can move below the curve without intersecting the curve as shown in figure 18c. This would also give us an idea about the largest link length that can be used in a hyper-redundant robot traversing a given duct. This concept can also be extended to tessellated surfaces with information about the principal curvatures of the surface at a point (e.g. see the work by Meyer et al. [??]).

# 6 Conclusions

Motion planning for hyper-redundant robots in narrow ducts using tractrix based redundancy resolution scheme is addressed. To this end, we discussed three methods to represent ducts in

[8]In backbone curve approach, this problem is addressed, but with the cost of very large displacement of the tail point.

[9]The relative curvature matrix of a parametric surface $S(u,v) : \mathbb{S}^2 \to \mathbb{R}^3$ and a sphere of radius R at the point of contact $P = \mathcal{S}(u^*, v^*)$ and on the Gaussian frame attached to the sphere(or $S$) at P is given as: $\kappa_R|_P = \begin{bmatrix} S_u.S_u - 1/R & S_u.S_v \\ S_u.S_v & S_v.S_v - 1/R \end{bmatrix}_{u^*,v^*}$




(a) Infeasible link length (b) Traversability of a curve (c) Feasible link lengths

Figure 18: Feasibility of motion planning with a given link length about a given curve.

2D plane as a combination of ellipses, as a series of connected quadrilaterals as well as a bound planar surface formed from two non-intersecting analytical curves. Methods to formulate inequality constraints which will impose the tail point of a link to always lie inside the duct are investigated for all the cases. In 3D, representation of ducts as series of ellipsoids, series of connected cylinders and using point clouds is discussed. The basic formulation as well as formulation for efficient practical implementation is discussed. The methods discussed are demonstrated using three practical examples of motion planning of 1) a hyper-redundant inspection robot through an industrial pipeline 2) a highly articulated endoscopic robot through a GI tract and 3) a hyper-redundant robot in search and rescue operation. From the examples, it is shown that the methods discussed in the paper can be effectively used in motion planning of a hyper-redundant robot manoeuvring in confined bounded spaces while emulating realism. Computational complexity for the methods discussed in this paper is analysed. The complexity of the methods scale linearly for the number of links as well as the number of path points for a given problem, while it scales with the order of maximum of 1.5 for the number of constraint equations. The locking condition for links in negotiating ducts of very low radius of curvature is discussed. A maximum link length which could overcome the issue, based on the curvature of the duct walls is suggested.

??uses LAPACK and heavy solvers.. not really required. instead a stripped version of the interior point will do quite well for fast near real-time implementation. ?? Research on algorithms to overcome the locking problem while maintaining motion realism is underway.

# 7 Appendix

## 7.1 Analytical expressions for $u, v$ for quadrilateral patch

$$v = \frac{k_1 - k_2 \pm \sqrt{(k_2 - k_1)^2 - 4k_3k_4}}{2k_3}, \quad u = \frac{(x - {}^xP_{i-1} + v)}{(a_2 + va_3)} \tag{31}$$

where, $k_1 = (b_1b_2 + b_0b_3)\,, \quad k_2 = (a_1a_2 + a_0a_3)\,, \quad k_3 = (a_1a_3 - b_1b_3)\,, \quad k_4 = (a_0a_2 - b_0b_2)$

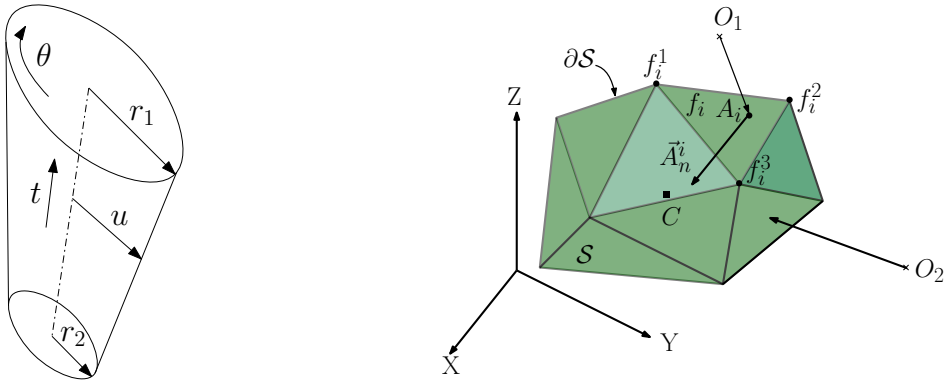
$$a_0 = y - {}^yP_{i-1},\ a_1 = {}^yP_{i-1} - {}^yP_i,\ a_2 = {}^xQ_{i-1} - {}^xP_{i-1},\ a_3 = ({}^xQ_i - {}^xP_i) - ({}^xQ_{i-1} - {}^xP_{i-1})$$

$$b_0 = x - {}^xP_{i-1},\ b_1 = {}^xP_{i-1} - {}^xP_i,\ b_2 = {}^yQ_{i-1} - {}^yP_{i-1},\ b_3 = ({}^yQ_i - {}^yP_i) - ({}^yQ_{i-1} - {}^yP_{i-1})$$

If ${}^xP_{i-1} = {}^xP_i$ and ${}^xQ_{i-1} = {}^xQ_i$,

$$u = \frac{b_0}{a_2}, \quad v = \frac{a_0a_2 - (a_3 + a_2)}{a_1\,(b_0 - a_2) + b_0\,(b_3 - b_2)} \tag{32}$$

## 7.2 Parametric equation of solid cylinder




(a) Parameters of a general cylinder

(b) Schematic of algorithm 1

Figure 19

The parametric cylinder, as shown in figure 19a, with the parameters $u,\ \theta$ and $t$ is given as:

$$x = C_1(u,t,\theta) = (r_1m_1u\cos\theta + m_2)\,(1-t) + (r_2n_1u\cos\theta + n_2)\,t \tag{33}$$

$$y = C_2(u,t,\theta) = (r_1m_3u\cos\theta + r_1m_4u\sin\theta + m_5)\,(1-t) + (r_2n_3u\cos\theta + r_2n_4u\sin\theta + n_5)\,t \tag{34}$$

$$z = C_3(u,t,\theta) = (r_1m_6u\cos\theta + r_1m_7u\sin\theta + m_8)\,(1-t) + (r_2n_6u\cos\theta + r_2n_7u\sin\theta + n_8)\,t \tag{35}$$

where

$$
\begin{aligned}
m_1 &= \cos{}^1\phi_2,\ m_2 = {}^1x_c, & m_3 &= \sin{}^1\phi_1 \sin{}^1\phi_2,\ m_4 = \cos{}^1\phi_1,\\
m_5 &= {}^1y_c,\ m_6 = -\cos{}^1\phi_1 \sin{}^1\phi_2, & m_7 &= \sin{}^1\phi_1,\ m_8 = {}^1z_c\\
n_1 &= \cos{}^2\phi_2,\ n_2 = {}^2x_c, & n_3 &= \sin{}^2\phi_1 \sin{}^2\phi_2,\ n_4 = \cos{}^2\phi_1,\\
n_5 &= {}^2y_c,\ n_6 = -\cos{}^2\phi_1 \sin{}^2\phi_2, & n_7 &= \sin{}^2\phi_1,\ n_8 = {}^2z_c
\end{aligned}
$$

The quantity ${}^1(\cdot)$ and ${}^2(\cdot)$ represent the corresponding parameters of the circles at the ends of the cylinder. $\phi_1$ and $\phi_2$ are the angles about the Y and X-axes which the plane of the circle is rotated, $(x_c, y_c, z_c)$ is the co-ordinate of the center of the circle.

## 7.3 An algorithm for classifying a point with respect to a polyhedron

Classification a point with respect to a triangulated domain is a very well known problem and many methods exist, which offer various advantages in terms of computational complexities. In algorithm 1, we describe a method, to classify a given set of points $O$ with respect to a polyhedron $\mathcal{P}$ as inside ($O_{in}$) and outside ($O_{out}$). The implementation of our algorithm[10] is $\mathcal{O}(m)$. The algorithm can be visualized from figure 19b.

**Purpose :** To classify a set of points $O$ with respect to boundary $\partial\mathcal{P}$
**Input:** The set $O$, $O \in \Re^3$ and $O \equiv O_{in} \cup O_{out}$
**Output:** $O_{in},\ O_{out}$

1: Obtain $C$, the center of $\mathcal{P}$ by taking the mean of the vertices of the $N$ facets constituting $\partial\mathcal{P}$
2: **for** $i = 1, 2, ..., N$ **do**
3: Calculate the normal to the $i^{th}$ facet, $A_n^i = (f_i^1 - f_i^2) \times (f_i^2 - f_i^3)$
4: Choose a point $A^i$ on the $i^{th}$ facet and move $A_n^i$ to $A^i$
5: Ensure that $A_n^i$ is directed inside, towards $C$
6: $H(i) = \langle O_i \vec{-} A_i, A_n^i \rangle$
7: **end for**
8: **if** $H(i) \geq 0\ \forall i$ **then**
9: Classify $O_1 \in O_{in}$ as inside, otherwise classify $O_1 \in O_{out}$
10: **end if**

**Algorithm 1:** Algorithm for classifying points as inside ($O_{in}$) or outside ($O_{out}$) of a triangulated domain.

[10] We have used a modified version of the Matlab function inhull.m, made available by John D'Errico for free usage.

# References


[1] A. Wolf, H. B. Brown, R. Casciola, A. Costa, M. Schwerin, E. Shamas, H. Choset, A mobile hyper redundant mechanism for search and rescue tasks, in: Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on, Vol. 3, IEEE, 2003, pp. 2889–2895.

[2] R. R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, A. M. Erkmen, Search and rescue robotics, in: Springer Handbook of Robotics, Springer, 2008, pp. 1151–1173.

[3] A. B. Slatkin, J. Burdick, W. Grundfest, The development of a robotic endoscope, in: Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on, Vol. 2, IEEE, 1995, pp. 162–171.

[4] R. D. Howe, Y. Matsuoka, Robotics for surgery, Annual Review of Biomedical Engineering 1 (1) (1999) 211–240.

[5] G.-B. Cadiere, J. Himpens, O. Germay, R. Izizaw, M. Degueldre, J. Vandromme, E. Capelluto, J. Bruyns, Feasibility of robotic laparoscopic surgery: 146 cases, World journal of surgery 25 (11) (2001) 1467–1477.

[6] J. Burgner-Kahrs, D. C. Rucker, H. Choset, Continuum robots for medical applications: A survey, IEEE Transactions on Robotics 31 (6) (2015) 1261–1280.

[7] E. Paljug, T. Ohm, S. Hayati, The jpl serpentine robot: a 12-dof system for inspection, in: Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on, Vol. 3, IEEE, 1995, pp. 3143–3148.

[8] G. S. Chirikjian, J. W. Burdick, Hyper-redundant robot mechanisms and their applications, in: Intelligent Robots and Systems' 91.'Intelligence for Mechanical Systems, Proceedings IROS'91. IEEE/RSJ International Workshop on, IEEE, 1991, pp. 185–190.

[9] R. L. William II, J. B. Mayhew IV, Obstacle-free control of the hyper-redundant nasa inspection manipulator, in: Proc. of the Fifth National Conf. on Applied Mechanics and Robotics, 1997, pp. 12–15.

[10] L. Gargiulo, P. Bayetti, V. Bruno, J.-C. Hatchressian, C. Hernandez, M. Houry, D. Keller, J.-P. Martins, Y. Measson, Y. Perrot, et al., Operation of an iter relevant inspection robot on tore supra tokamak, Fusion Engineering and Design 84 (2-6) (2009) 220–223.

[11] S. Ma, S. Hirose, H. Yoshinada, Development of a hyper-redundant multijoint manipulator for maintenance of nuclear reactors, Advanced robotics 9 (3) (1994) 281–300.

[12] S. Hirose, M. Mori, Biologically inspired snake-like robots, in: Robotics and Biomimetics, 2004. ROBIO 2004. IEEE International Conference on, IEEE, 2004, pp. 1–7.

[13] M. W. Hannan, I. D. Walker, Analysis and initial experiments for a novel elephant's trunk robot, in: Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on, Vol. 1, IEEE, 2000, pp. 330–337.

[14] G. S. Chirikjian, J. W. Burdick, Design and experiments with a 30 dof robot, in: Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on, IEEE, 1993, pp. 113–119.

[15] A. Liegeois, Automatic supervisory control of the configuration and behaviour of multibody mechanisms, IEEE Transactions on systems, man and cybernetics 7 (12) (1977) 868–871.

[16] J. Baillieul, J. Hollerbach, R. Brockett, Programming and control of kinematically redundant manipulators, in: Decision and Control, 1984. The 23rd IEEE Conference on, Vol. 23, IEEE, 1984, pp. 768–774.

[17] A. A. Maciejewski, C. A. Klein, Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments, The international journal of robotics research 4 (3) (1985) 109–117.

[18] Y. K. Hwang, N. Ahuja, Gross motion planning—a survey, ACM Computing Surveys (CSUR) 24 (3) (1992) 219–291.

[19] S. Chiaverini, Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators, IEEE Transactions on Robotics and Automation 13 (3) (1997) 398–410.

[20] L. Sciavicco, B. Siciliano, A solution algorithm to the inverse kinematic problem for redundant manipulators, IEEE Journal on Robotics and Automation 4 (4) (1988) 403–410.

[21] Z. Mao, T. C. Hsia, Obstacle avoidance inverse kinematics solution of redundant robots by neural networks, Robotica 15 (1) (1997) 3–10.

[22] B. Dasgupta, A. Gupta, E. Singla, A variational approach to path planning for hyper-redundant manipulators, Robotics and Autonomous Systems 57 (2) (2009) 194–201.

[23] G. Antonelli, S. Chiaverini, Fuzzy redundancy resolution and motion coordination for underwater vehicle-manipulator systems, IEEE Transactions on Fuzzy Systems 11 (1) (2003) 109–120.

[24] V. J. Lumelsky, A. A. Stepanov, Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape, Algorithmica 2 (1-4) (1987) 403–430.

[25] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, in: Autonomous robot vehicles, Springer, 1986, pp. 396–404.

[26] H. Choset, Coverage for robotics–a survey of recent results, Annals of mathematics and artificial intelligence 31 (1-4) (2001) 113–126.

[27] S. Sreenivasan, P. Goel, A. Ghosal, A real-time algorithm for simulation of flexible objects and hyper-redundant manipulators, Mechanism and Machine Theory 45 (3) (2010) 454–466.

[28] M. S. Menon, G. Ananthasuresh, A. Ghosal, Natural motion of one-dimensional flexible objects using minimization approaches, Mechanism and Machine Theory 67 (2013) 64–76.

[29] M. S. Menon, V. Ravi, A. Ghosal, Trajectory planning and obstacle avoidance for hyper-redundant serial robots, Journal of Mechanisms and Robotics 9 (4) (2017) 041010.

We have to see IEEE reference format.