2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007



# Limiting-case Analysis of Continuum Trunk Kinematics

Bryan A. Jones, *Member, IEEE* and Ian D. Walker, *Fellow, IEEE*

***Abstract*—Continuum robotic manipulators, termed trunks, mimic the astounding capabilities of elephant trunks and octopus arms by bending in smooth arcs. Several approaches to kinematic analysis of continuum trunks complement a wide variety of available continuum robots. However, these kinematics exhibit singularity-like conditions when the trunk assumes a straight posture, which is essential to complete many tasks. The novel limiting-case analysis presented in this paper eliminates these problems, demonstrating that the unique causes of the problem are rooted in the continuum formulation and cannot be solved by traditional rigid-link singularity analysis. Three practical examples demonstrate the necessity of this analysis presented, enabling the trunk to successfully perform each task.**

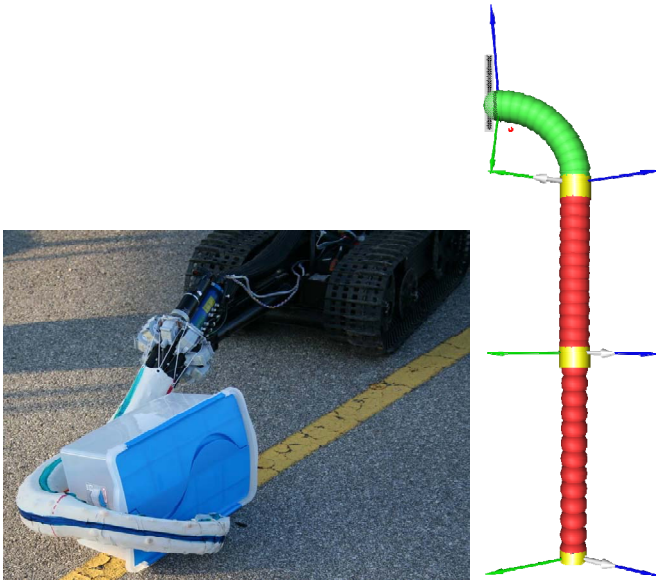
## I. Introduction

THE unparalleled dexterity of biological structures free of skeletal constraints remain a marvel of dexterity and possess an amazing range of abilities. An elephant's trunk provides sufficient force to grasp and maneuver large tree trunks while providing the agility and gentleness to pick up a peanut. The arms of an octopus provide the creature with a means of propulsion, prey capture, environmental exploration, and self defense. Tongues likewise enable mastication, prey capture in the case of some reptiles, cleaning, and sensing. Continuum robots seek to replicate these remarkable abilities by incorporating flexible materials in their construction, yielding structures which bend in continuous curves. A number of commercial and research robots overviewed in [1] provide many mechanical platforms with which to realize these possibilities, and ongoing work in kinematic [2, 3, 4, 5, 6, 7] and dynamic [7, 8, 9] analysis of these structures supplies a mathematical basis to specify the desired shape of these robots. The kinematic formulations referenced above model the manipulator, henceforward termed a trunk, as a serially connected series of arc segments each possessing some radius $r$.

One immediate difficulty typified in Fig. 1 and shared by all these formulations commonly occurs when one or more of the sections of the trunk do not bend, in which case the bending radius of the trunk section becomes $\infty$. This gives rise to two problems. First, numerical evaluation of the kinematics at this point then inevitably involves terms including $r$, resulting in undefined results. However, mathematically evaluating $\lim_{r\to\infty}$ produces expressions which can then be numerically evaluated to produce correct results. Second, evaluation near the limiting case of a straight trunk in finite-precision machine arithmetic produces numerical instability. These problems, heretofore unaddressed to the best of the author's knowledge, render the kinematics uncomputable in the common unbent trunk configuration and produce unexpected results when trunk trajectories pass through or near this configuration.

This paper presents a careful analysis of the kinematic equations of continuum trunks, combining in a single expression the ability to accurately compute the kinematics away from, at, or near these limiting cases in finite-precision machine arithmetic. Analysis of this approach demonstrates its correctness when compared with mathematically evaluating expressions at $\lim_{r\to\infty}$. This novel technique is therefore essential when computing real-time practical kinematics for a trunk and can be broadly applied to all the works cited above.

Several kinematic approaches exist in the literature. Modeling the movement of snakes [10] produced continuum-like kinematics in 2-D. An alternate approach involves choosing



(a) (b)

Fig. 1. Common straight configurations which require a limiting-case analysis. In (a), the first section of the OctArm continuum robot maintains a straight posture to perform a grasping task. Image (b) shows a simulated continuum trunk with two straightened sections, enabling it to reach a workspace boundary.

This work was supported in part by the Defense Advanced Research Projects Agency (DARPA), Contract Number N66001-C-8043.

Bryan A. Jones is with the Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS 39762 USA (phone: 662-325-3149; fax: 662-325-9438; e-mail: bjones @ ece.msstate.edu).

Ian D. Walker is with the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC 29634 USA (e-mail: ianw @ ece.clemson.edu).

a mathematically convenient curve then fitting a trunk to it [11], often an imprecise approximation for continuum robots. Examining the physics of flexible structures demonstrates that continuum trunks bend with constant curvature, forming an arc of a circle, in the absence of external forces such as gravity [8]. This insight yielded several kinematic formulations [2, 3, 4, 5, 6, 7] which convert either: trunk shape, measured in a direction of curvature $\phi$, amount of curvature $\kappa = 1/r$ defined as the inverse of the radius of the arc, and arc length $s$; actuator lengths $l_{1-3}$; or per-actuator pressures for pneumatic trunks to trunk tip coordinates.

Previous approaches to the problem of an infinite trunk radius when the trunk is straight include requiring the trunk to be bent at all times [4], though most papers simply do not address the problem. Classical rigid-link robotic analysis provides little help; indeed, it lacks even a common terminology for the problem. Singularities formally describe configurations in which the Jacobian of a manipulator loses rank. However, in the case of a continuum robot the rank of the Jacobian does not change when the trunk is straight. Instead, the Jacobian becomes undefined due to limiting cases embedded within it, thus motivating the title of this work. Careful examination of these cases provides methods to correctly work around this problem. As a specific case, this paper applies these techniques to the kinematics in [2].

## II. Limiting-case analysis

Work in [1] classifies continuum robots in three types: intrinsic, extrinsic, and hybrid. Because each trunk section possesses no more than three degrees of freedom, most hardware prototypes choose a design featuring three actuators. Intrinsic designs [2, 4, 5, 6, 7] produce movement by varying pressures in three flexible tubes while extrinsic and hybrid trunks [12, 13, 14] use three cables to determine shape. Kinematics developed in [2] provide expressions for both intrinsic and extrinsic/hybrid trunks, covering the majority of the hardware designs available.

As derived in [2], the length $\underline{l}$ of three actuators determines the shape of one section of the trunk. A review of the equations used to derive forward kinematics and forward velocity kinematics reveals multiple instances of the actuator length squared difference term

$$g \triangleq l_1^2 + l_2^2 + l_3^2 - l_1 l_2 - l_2 l_3 - l_1 l_3. \tag{1}$$

For example, the an entry from the homogenous transformation matrix $\mathbf{A}$ which maps actuator lengths $l_{1-3}$ to a translation and rotation of the trunk tip [2] is

$$\mathbf{A}_{1,1} = 1 - \frac{3(l_3 - l_2)^2}{2g}\left(\sin^2\left(\frac{\sqrt{g}}{3d}\right)\right) \tag{2}$$

where $l_{1-3}$ gives the three actuator lengths and $d$ defines the radius of the trunk's cross-section. The remaining elements of $\mathbf{A}$ are given in (8). The Jacobian $\mathbf{J}$ which maps actuator velocities to tip velocities likewise contains the $g$ term [2]. Only one element of the lengthy $\mathbf{J}$ matrix is given here,

$$\mathbf{J}_{6,1} = \frac{\sqrt{3}(l_3 - l_2)\sin^2\left(\frac{\sqrt{g}}{3d}\right)}{g}. \tag{3}$$

These equations can be analytically evaluated at their limiting case when $g \to 0$. As shown in the appendix, $g$ is proportional to the distance of a point $\begin{bmatrix} l_1 & l_2 & l_3 \end{bmatrix}^{\mathrm{T}}$ in 3-D space from the line $l_1 = l_2 = l_3$. Assuming actuator lengths $l_{1-3}$ to be real, $g = 0 \Leftrightarrow l_1 = l_2 = l_3$. Therefore, evaluation of equation (2) at its limiting case of $\lim_{g\to 0} \mathbf{A}_{1,1} = 1 - \frac{3(l_3 - l_2)^2}{2g}\left(\frac{\sqrt{g}}{3d}\right)^2 = 1 - \frac{(l_3 - l_2)^2}{6d^2} = 1$, noting that $\lim_{x\to 0} \sin x = x$ and that $l_2 = l_3$ because $g = 0$. Similar applications to the remainder of the $\mathbf{A}$ matrix yield

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

where $l \triangleq l_1 = l_2 = l_3$. That is, when all three actuators are of equal length $l$, the trunk extends along the $+z$ axis by $l$.

Likewise, examination of $\mathbf{J}$ when evaluated as $g \to 0$ reveals $\lim_{g\to 0} \mathbf{J}_{6,1} = \frac{\sqrt{3}(l_3 - l_2)}{g}\left(\frac{\sqrt{g}}{3d}\right)^2 = \frac{\sqrt{3}(l_3 - l_2)}{9d^2} = 0$ as the actuator length squared difference term $g$ approaches zero, again recalling that $\lim_{x\to 0} \sin x = x$ and that $g = 0 \Leftrightarrow l_1 = l_2 = l_3$. Evaluating the remainder of the $\mathbf{J}$ matrix gives

$$\mathbf{J} = \begin{bmatrix} 0 & l\sqrt{3}/6d & -l\sqrt{3}/6d \\ -l/3d & l/6d & l/6d \\ 1/3 & 1/3 & 1/3 \\ 2/3d & -1/3d & -1/3d \\ 0 & \sqrt{3}/3d & -\sqrt{3}/3d \\ 0 & 0 & 0 \end{bmatrix}. \tag{5}$$

As a result, there are two expressions for the $\mathbf{A}$ matrix and the $\mathbf{J}$ matrix, depending on the equality of the actuator length $\underline{l}$. When $l_1 = l_2 = l_3$, these matrices are given in (4) and (5); otherwise, they are non-linear functions which depend on $\underline{l}$ as given in [2] and in (8). This leads to a number of problems when attempting to evaluate $\mathbf{A}$ or $\mathbf{J}$ at a specific configuration of the trunk. First, numerical accuracy problems resulting from the use of finite-precision machine arithmetic arise in a region of the configuration space as the

$$\mathbf{A} = \begin{bmatrix} 1-\frac{3(l_3-l_2)^2(1-c_g)}{4g} & \frac{\sqrt{3}(l_3-l_2)(l_2+l_3-2l_1)(1-c_g)}{4g} & -\frac{\sqrt{3}(l_3-l_2)s_g}{2g} & -\frac{d\sqrt{3}(l_3-l_2)(l_1+l_2+l_3)(1-c_g)}{4g} \\ \frac{\sqrt{3}(l_3-l_2)(l_2+l_3-2l_1)(1-c_g)}{4g} & c_g+\frac{3(l_3-l_2)^2-(3l_2^2+3l_3^2-6l_2l_3)c_g}{4g} & \frac{(l_2+l_3-2l_1)s_g}{2\sqrt{g}} & -\frac{d\sqrt{3}(l_2+l_3-2l_1)(l_1+l_2+l_3)(1-c_g)}{4g} \\ \frac{\sqrt{3}(l_3-l_2)s_g}{2g} & -\frac{(l_2+l_3-2l_1)s_g}{2\sqrt{g}} & c_g & \frac{d(l_1+l_2+l_3)s_g}{2\sqrt{g}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

actuator length squared difference term $g$ tends toward zero. Further analysis is required to determine the size of this region, which depends on the accuracy of the finite-precision arithmetic used. Second, when using the limiting case analysis, solutions for each section of the trunk require two formulas, one for the typical case and a second to handle the limiting case. In an $n$-section trunk, this results in the need for $2^n$ equations to handle all combinations of limiting and non-limiting cases for all sections. Third, finding and analyzing these cases, in which limiting cases from earlier, proximal sections will couple into the equations for later, distal sections, is a daunting task. Consider, for example, how expression (3), one matrix element in a single-section Jacobian, must grow when computing a four-section Jacobian (corresponding to the hardware in [3]) which requires $2^4 = 16$ sets of equations, all of which will contain fractional powers of the actuator length squared difference term inside sin, $\sin^{-1}$, cos, and $\cos^{-1}$ functions which much be analyzed for limiting cases. These three problems indicate the need for an alternative, comprehensive analysis which, in one expression, provides accurate numerical results for finite-precision arithmetic at or near the limiting case for a multi-section trunk.

### *A. Error metric*

Therefore, this work pursues a simpler approach, though one which requires additional off-line analysis. Instead of employing the limiting-case expressions outlined in the previous section, the original equations defining **A** or **J** can be evaluated near, but not at, the limiting case. This approach both solves the problem of evaluating these expressions at their limiting point, which would otherwise produce a divide-by-zero floating-point exception, and numerical instabilities near the limiting point due to finite precision effects.

To perform such an analysis requires a definition of error and method of computing it. A straightforward error metric would be to examine

$$\left\| \mathbf{M}_{machine} - \mathbf{M}_{exact} \right\| \quad (6)$$

where **M** is either **A** or **J**, $\mathbf{M}_{machine}$ is the matrix evaluated in finite-precision machine arithmetic, and $\mathbf{M}_{exact}$ gives the exact value of the matrix. However, computing the desired matrix involves evaluating trigonometric functions and fractional powers, precluding computation of an exact value. Instead, a more practical approach is to use a high, but finite-precision result for $\mathbf{M}_{exact}$. To evaluate the accuracy of $\mathbf{M}_{exact}$, recall the limits evaluated earlier of the form $\lim_{g\to 0} \mathbf{M}(l_1, l_2, l_3) = \mathbf{C}$, which states that for a given error $\varepsilon > 0$ there exists a perturbation $\Delta > 0$ such that

$$\left\| \mathbf{M}(l_1 + \Delta, l_2, l_3) - \mathbf{C} \right\| < \varepsilon, \quad (7)$$

where **C** is the analytically-determined limit of the form given in (4) and (5). This inequality holds under accurate numerical evaluation and fails otherwise, providing a test to determine the valid region for a finite-precision calculation.

For example, consider Fig. 2, showing the evaluation of **J** at 60 decimal digits of precision using Maple where trunk parameters are $d = 1$ and $l_1 = l_2 = l_3 = 10$. This graph demonstrates that (7) cannot be satisfied due to numerical accuracy problems when $\Delta < 10^{-19}$. Therefore, we deem this **J** "exact" outside this region, when $\Delta > 10^{-19}$, and proceed with examination of the error defined by (6).

This clear understanding of error then enables an analysis of the numerical performance of the **A** and **J** matrices near their limiting point, as discussed in the following section.

### *B. Comprehensive analysis*

The analysis begins with the essential observation that all limiting cases in both **A** and **J** occur only when the actuator lengths are equal (resulting in a locally straight trunk), due to terms of the form $1/g$, where $g$ is defined in (1), which tends to zero in this case. As given in (1), computing $g$ involves the sum and difference of the product of actuator lengths. Rewriting as

$$g = (l_2 - l_1)^2 + (l_3 - l_1)^2 - (l_2 - l_1)(l_3 - l_1) \quad (9)$$

instead evaluates $g$ in terms of the squared differences between actuator lengths. For example, using IEEE 754 double-precision floating-point arithmetic [15] which provides 53 bits or ~16 decimal digits of precision, at the point $l_1 = 10 + \Delta$ and $l_2 = l_3 = 10$ any value for which $\Delta < 1.01 \cdot 10^{-7}$ produces $g = 0$ when computing $g$ using (1) while a much smaller $\Delta < 8.89 \cdot 10^{-16}$ produces $g = 0$ when computing $g$ using (9). To prevent the $g = 0$ condition and consequent

$1/g = 1/0$ calculations, a small perturbation of $2.26 \cdot 10^{-308}$ (the smallest representable double-precision floating-point number) can be added to the $g$ computed in (9) without affecting the resulting accuracy of the computations.

In the case of the **A** matrix, careful hand analysis of the terms involved leads to the expression given in (8), where $g$ is defined in the paragraph above and includes a tiny additive perturbation, $d$ gives the distance from the trunk center to an actuator center (the "radius of actuation"), $c_g = \cos\left(2\sqrt{g}/3d\right)$, and $s_g = \sin\left(2\sqrt{g}/3d\right)$. In the matrix, first note that terms such as $(l_3 - l_2)$ and $(l_2 + l_3 - 2l_1)$ approach 0 as $g$ goes to 0. This causes most off-diagonal terms to become 0 and diagonal terms to tend to 1, agreeing with the results in (4), since $c_g = \cos 0 = 1$ in this case. The exception to this trend is $\mathbf{A}_{3,4}$, whose numerical properties yield the desired limiting-case results of $l$ when $g \to 0$. Therefore, the analysis given in the previous section shows that (7) evaluated with $\mathbf{M} = \mathbf{A}$ can be satisfied for all actuator lengths in double-precision floating point arithmetic, obviating need for further high-precision (60 decimal digit) analysis. The **A** matrix for a multi-section trunk is the product of the per-section **A** matrices, providing numerically stable and accurate results for any number of sections.

However, the additional complexity of the Jacobian matrix makes hand analysis to achieve results similar to the **A** matrix extremely difficult. Instead, numerical instabilities in the region around $l_1 = l_2 = l_3$ require evaluation outside this region for accurate results. Simply adding a small offset to one of the actuator lengths as necessary to remain outside the region of instability avoids these limiting-case difficulties. More formally, we seek a minimum value $\delta$ for $g$ which avoids numerical problems by guaranteeing that $g$ never falls below $\delta$ by requiring

$$g = l_1^2 + l_2^2 + l_3^2 - l_1 l_2 - l_2 l_3 - l_1 l_3 > \delta. \tag{10}$$

Solving for $l_1$, this minimum value is violated when

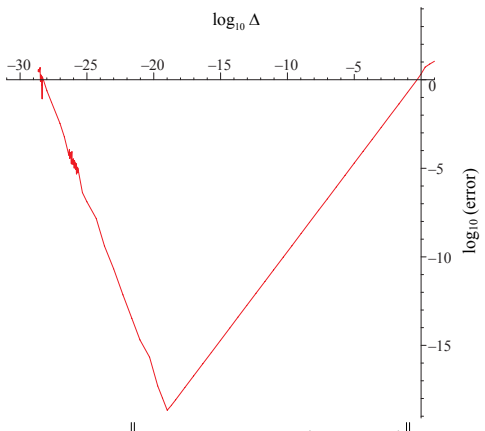



Fig. 2. A plot of $error = \left\| \mathbf{J}(l_1 + \Delta, l_2, l_3) - \lim_{g \to 0} \left( \mathbf{J}(l_1, l_2, l_3) \right) \right\|_F$, showing that the high-precision (to 60 decimal digits) Jacobian accurately approximates the analytically-determined limiting-case Jacobian for $\Delta > 10^{-19}$.

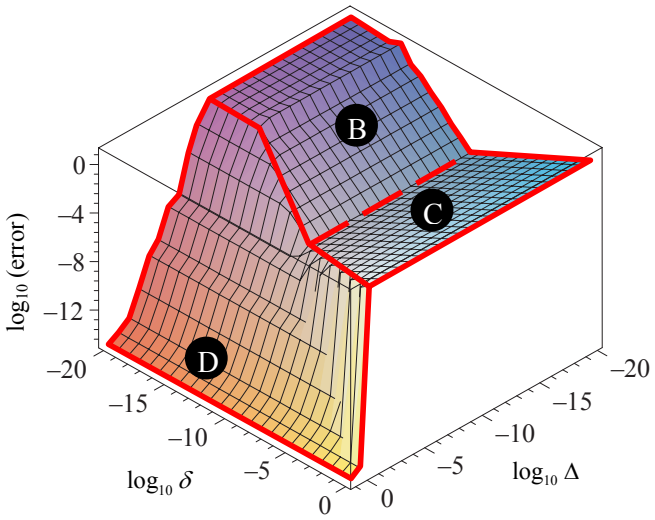



Fig. 3. Comparison of the error $\left\| \mathbf{J}_{60}(l_1 + \Delta, l_2, l_3) - \mathbf{J}_{16}\left(l_{1p}(l_1 + \Delta, l_2, l_3, \delta), l_2, l_3\right) \right\|_F$ over a range of differences $\Delta$ in actuator length for a two-section trunk as the minimum allowable actuator squared length difference term $\delta$ changes. The use of $l_{1p}$ as defined by (12) guarantees this minimum.

$$\begin{aligned} &\frac{l_2 + l_3 - \sqrt{6l_2l_3 + 4\delta - 3l_2^2 - 3l_3^2}}{2} < l_1 \text{ and} \\ &l_1 < \frac{l_2 + l_3 + \sqrt{6l_2l_3 + 4\delta - 3l_2^2 - 3l_3^2}}{2}. \end{aligned} \tag{11}$$

Therefore, define a perturbed actuator length $l_{1p}$ such that

$$l_{1p} = \begin{cases} \dfrac{l_2 + l_3 \pm \sqrt{6l_2l_3 + 4\delta - 3l_2^2 - 3l_3^2}}{2} & \text{when (7)} \\ & \text{is satisfied} \\ l_1 & \text{otherwise.} \end{cases} \tag{12}$$

The parameter $\delta$ should be chosen so that it precisely encompasses the region around the limiting point $l_1 = l_2 = l_3$ where machine-precision floating-point inaccuracies would produce 1/0 results or evaluating terms of the form $f(g)/g$, such as (2) and (3), would produce larger errors in **A** or **J** compared to evaluation at the perturbed point. To measure the size of this region, consequently determining an appropriate value for $\delta$, consider the use of (6) when $\mathbf{M} = \mathbf{J}$ and $\mathbf{J}_{exact} = \mathbf{J}_{60}$, where $\mathbf{J}_i$ indicates a Jacobian computed with $i$ decimal digits of precision.

Fig. 3 shows the resulting error generated by evaluating $\left\| \mathbf{J}_{60}(l_1 + \Delta, l_2, l_3) - \mathbf{J}_{16}\left(l_{1p}(l_1 + \Delta, l_2, l_3, \delta), l_2, l_3\right) \right\|_F$ where $l_{1p}$ is defined by (12). Because IEEE 754 double-precision floating-point arithmetic [15] specifies ~16 decimal digits of accuracy, $\mathbf{J}_{16}$ approximates a machine-precision computation. These results were obtained by choosing trunk radius $d = 1$ and $l_1 = l_2 = l_3 = 10$ for a one-section pneumatically-driven (intrinsically actuated) trunk. The figure was generated by injecting a range of actuator length differences of $\Delta$ into $l_1$ while also varying the minimum allowed actuator

length squared difference $\delta$.

The figure can be divided into three regions. In region D, $l_{1p} = l_1$. Therefore, only minor differences are produced when comparing the machine-precision and high-precision results. However, as $\Delta$ continues decreasing, the resulting actuator length squared difference $g$ drops below its minimum of $\delta$, splitting the results into two parts. In region C, large values of $\delta$ inject an unnecessarily large perturbation into the resulting $l_{1p}$ used by the machine-precision Jacobian evaluation, producing error constant under changing $\Delta$, since the limit $\delta$ enforces an injection of a perturbation $> \Delta$, varying only as $\delta$ changes. In region B, overly small values of $\delta$ allow finite-precision affects to unnecessarily increase the error produced when evaluating Jacobian too close to its limiting point.

As indicated by the dotted line in the figure between regions B and C, an optimal value for $\delta$ is $\delta = 10^{-10}$, yielding a maximum error of $e < 10^{-2}$ by injecting a perturbation of $3.1623 \cdot 10^{-4}$ in actuator 1's length when the actuator length squared difference term is less than $\delta = 10^{-10}$. Setting $\delta > 10^{-10}$ perturbs actuator lengths more than necessary, reducing the accuracy of the results, while selecting $\delta < 10^{-10}$ produces greater error or even undefined results due to finite-precision effects. Analysis of a one-section cable-driven (hybrid actuation) trunk and of a two-section trunk, shown in Fig. 4 and in Fig. 5, reveals similar limits apply to these cases.

Therefore, the following algorithm insures accurate computation of the Jacobian, avoiding floating-point exceptions when actuator lengths are equal and numerical instabilities when actuator lengths are almost equal. The algorithm computes the value of $l_{1p}$ by evaluating (12). That is, given three actuator lengths $l_1$, $l_2$, and $l_3$ with which the Jacobian should be computed, the algorithm produces a possibly perturbed actuator length $l_{1p}$ and unperturbed lengths $l_2$, and $l_3$ with which the Jacobian can be computed free of numerical problems. The procedure to produce $l_{1p}$ is:

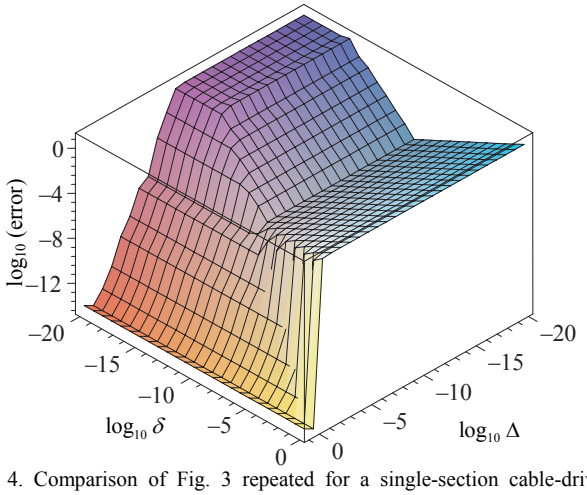



Fig. 4. Comparison of Fig. 3 repeated for a single-section cable-driven (hybrid actuation) trunk where the number of segments in each trunk section is $n = 1$.

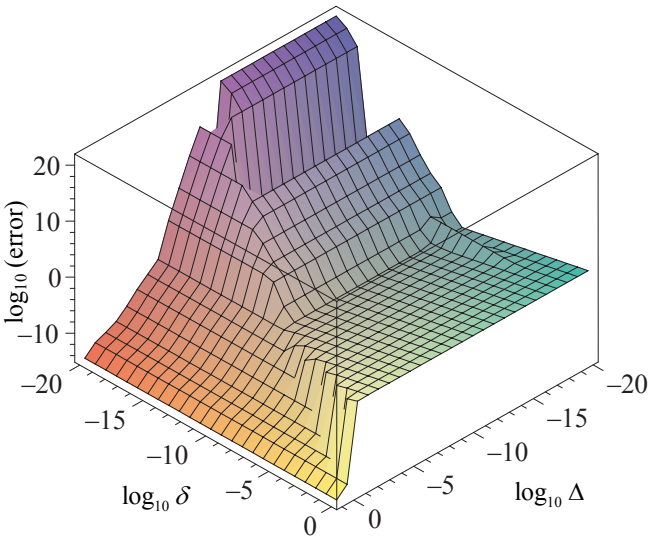



Fig. 5. Comparison of Fig. 3 repeated for a two-section trunk.

1. Evaluate the inequality given in (10).
   If $l_1^2 + l_2^2 + l_3^2 - l_1 l_2 - l_2 l_3 - l_1 l_3 > \delta$, set $l_{1p} \leftarrow l_1$.
   If $l_1^2 + l_2^2 + l_3^2 - l_1 l_2 - l_2 l_3 - l_1 l_3 \le \delta$, set
   $l_{1p} \leftarrow \dfrac{l_2 + l_3 \pm \sqrt{6 l_2 l_3 + 4\delta - 3 l_2^2 - 3 l_3^2}}{2}$, choosing the $l_{1p}$ closest to $l_1$.
2. Compute $\mathbf{J}\left(l_{1p}, l_2, l_3\right)$.

In summary, the technique of perturbing actuator lengths when these lengths are equal by replacing $l_1$ with $l_{1p}$ as defined in (12) successfully avoids both divide-by-zero floating-point exceptions when the actuator lengths are equal and numeric instabilities when actuator lengths are almost equal.

This technique enables real-time evaluation of multi-section Jacobians for continuum trunks, avoiding the many difficulties associated with a liming-case analysis. However, this approach requires careful choice of the minimum permissible actuator length squared difference term of $\delta = 10^{-10}$ as demonstrated by Fig. 3 and discussed in the preceding paragraphs. A similar approach and analysis can also be applied to computing Jacobian and **A** matrices for trunks defined by an direction bending $\phi$, amount of curvature $\kappa$, and trunk length $s$, making the approach applicable to continuum robots based on this parameterization.

## III. Applications

Three varying tasks illustrate the necessity of a straightened trunk to perform a variety of useful tasks. First, insertion tasks both begin and end with a straight trunk; bending may cause insertion or removal failure and possibly damage to the trunk. Fig. 1 illustrates the remaining two tasks, in which a straight posture is required for grasping and in order to access the workspace boundaries. Each task is detailed below.

Whether cleaning the interior of hazardous waste tanks or inspecting the interior of aircraft turbines, a typical task for

many trunk-like robots consists of inserting the trunk through a small opening then manipulating objects behind the opening. Critical to task completion is the insertion process, in which the trunk must first be completely straightened. Next, a prismatic joint anterior to the trunk performs the insertion. The trunk then moves from its straight insertion posture to perform the desired tasks, after which it returns to the straight configuration and is removed via the prismatic joint.

When operating a multi-section trunk near its workspace boundary, one or more trunk sections must often move to their straight configuration to provide the maximum extension necessary to reach the workspace boundary. Consider Fig. 1(b), in which two sections of a three-section trunk are straight. Any curvature in these sections would shorten the trunk, reducing its workspace, demonstrating the necessity for this analysis in order to fully utilize the robot's workspace.

As a last example, consider Fig. 1(a). Here, object placement may require a straight trunk configuration in order to accomplish the desired task of grasping then manipulating a plastic storage box. Other configurations may not place the trunk in a posture enabling it to grasp the box.

Because this work applies to any continuum robot modeled by a circular arc, it applies to many additional problems when the trunk assumes a straight posture. For example, the dynamic model given in [16], which is based on the circular arc assumption, can be easily extended to accommodate a straight trunk by applying the techniques suggested in this paper.

## IV. Conclusion

The wide variety of continuum robots provide a remarkable set of manipulator abilities. Several varieties of kinematic formulations provide the ability to move and shape the trunk as desired. However, these approaches all suffer from the same problem of numerical instability or even undefined results when a continuum trunk enters or nears a straight configuration, which is essential for performing a wide variety of tasks. Analysis of the kinematics in this paper provides a novel method of avoiding these problems by properly factoring terms in the positional kinematics or by carefully perturbing actuator lengths the minimum amount necessary to produce stable results for the velocity kinematics. An error metric allows examination of the resulting computations and demonstrates their accuracy and validity.

## Appendix

As claimed in section II, $g$ is proportional to the distance of a point $\underline{l} = \begin{bmatrix} l_1 & l_2 & l_3 \end{bmatrix}^{\mathrm{T}}$ in 3-D space from the line $l_1 = l_2 = l_3$. To prove this assertion, first project the point onto the line. Choosing any vector which lies along the line, such as $\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^{\mathrm{T}}$, the projection is defined by

$$\frac{\underline{l} \cdot \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^{\mathrm{T}}}{\left\| \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^{\mathrm{T}} \right\|_2^2} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^{\mathrm{T}} = \left( \frac{l_1 + l_2 + l_3}{3} \right) \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^{\mathrm{T}}.$$

Subtracting this vector from $\underline{l}$ yields the component of $\underline{l}$ perpendicular to the line $\underline{l} = (l_1, l_2, l_3)$. The length of this perpendicular component gives the distance from a point $\underline{l} = \begin{bmatrix} l_1 & l_2 & l_3 \end{bmatrix}^{\mathrm{T}}$ in 3-D space from the line $l_1 = l_2 = l_3$. Using the norm to measure this distances gives

$$\left\| \begin{bmatrix} l_1 & l_2 & l_3 \end{bmatrix}^{\mathrm{T}} - \tfrac{1}{3}(l_1 + l_2 + l_3) \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^{\mathrm{T}} \right\|_2 = g\sqrt{6}/3,$$

proving the assertion.

## References


[1] G. Robinson and J. B. C. Davies, "Continuum robots - a state of the art," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Detroit, Michigan, 1999, pp. 2849-2854.

[2] B. A. Jones and I. D. Walker, "Kinematics for Multisection Continuum Robots," *IEEE Transactions on Robotics,* vol. 22, pp. 43-55, Feb. 2006.

[3] M. W. Hannan and I. D. Walker, "Kinematics and the Implementation of an elephant's trunk manipulator and other continuum style robots," *Journal of Robotic Systems,* vol. 20, pp. 45-63, Feb. 2003.

[4] Y. Bailly and Y. Amirat, "Modeling and Control of a Hybrid Continuum Active Catheter for Aortic Aneurysm Treatment," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Orlanda, FL, USA, 2006, pp. 936-941.

[5] G. Chen, P. M. Tu, T. R. Herve, and C. Prelle, "Design and modeling of a micro-robotic manipulator for colonoscopy," in *5th Intl. Workshop on Research and Education in Mechatronics*, Annecy, France, 2005, pp. 109-114.

[6] F. Thomann, M. Betemps, and T. Redarce, "The development of a bendable colonoscopic tip," in *International Conference on Robotics and Automation*, Taipei, Taiwan, 2003, pp. 658-663.

[7] K. Suzumori, S. Iikura, and H. Tanaka, "Applying a flexible microactuator to robotic mechanisms," *Control Systems Magazine, IEEE,* vol. 12, pp. 21-27, 1992.

[8] I. A. Gravagne, C. D. Rahn, and I. D. Walker, "Large deflection dynamics and control for planar continuum robots," *IEEE/ASME Transactions on Mechatronics,* vol. 8, pp. 299-307, June 2003.

[9] M. Ivanescu, N. Popescu, and D. Popescu, "A Variable Length Tentacle Manipulator Control System," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005, pp. 3274-3279.

[10] S. Hirose, *Biologically inspired robots*: Oxford University Press, 1993.

[11] G. S. Chirikjian and J. W. Burdick, "A modal approach to hyper-redundant manipulator kinematics," *IEEE Transactions on Robotics and Automation,* vol. 10, pp. 343-354, June 1994.

[12] R. Buckingham, "Snake arm robots," *Industrial Robot: An International Journal,* vol. 29, pp. 242-245, 2002.

[13] N. Simaan, "Snake-Like Units Using Flexible Backbones and Actuation Redundancy for Enhanced Miniaturization," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005, pp. 3023-3028.

[14] T. Aoki, A. Ochiai, and S. Hirose, "Study on slime robot: development of the mobile robot prototype model using bridle bellows," in *Proceedings of the IEEE International Conference on Robotics and Automation*, New Orleans, Louisiana, 2004, pp. 2808-2813.

[15] *IEEE Standard for Binary Floating-Point Arithmetic*: ANSI/IEEE Std 754-1985.

[16] E. Tatlicioglu, I. D. Walker, and D. M. Dawson, "Dynamic Modelling for Planar Extensible Continuum Robot Manipulators," in *International Conference on Robotics and Automation* Rome, Italy, 2007.