A Control Framework for Snake Robot Locomotion based on Shape Control Points Interconnected by Bézier Curves

Pål Liljebäck, Kristin Y. Pettersen, Øyvind Stavdahl, and Jan Tommy Gravdahl

***Abstract*— This paper presents a control framework for shape control of snake robots for the purpose of locomotion. An advantage of the framework is that it allows the macroscopic shape of a snake robot to be controlled explicitly and intuitively. The framework is based on specifying the desired shape of the snake robot as a continuous *shape curve* defined by a set of *shape control points* interconnected by *Bézier curves*. We propose a novel approach for motion generation in which the shape curve is repeatedly extended according to a desired gait pattern while a virtual snake robot is progressed along the shape curve to retrieve joint reference angles for the physical snake robot. Practical applications of the proposed control framework are exemplified along with simulation results.**

I. Introduction

Inspired by biological snakes, snake robots carry the potential of meeting the growing need for robotic mobility in challenging environments. Snake robots consist of serially connected modules capable of bending in one or more planes. The many degrees of freedom of snake robots make them difficult to control, but provide traversability in irregular environments which surpasses the mobility of more conventional wheeled, tracked and legged robots.

Locomotion of a snake robot is achieved by body shape changes which cause the body to interact with its environment and propel the robot in some direction. The literature contains numerous approaches for shape control of snake robots. Some approaches mainly consider the kinematics of the snake robot [1]–[5], while other approaches also take the dynamics of the locomotion into account [6]–[9]. The majority of previous approaches for motion control of snake robots carry out shape control by specifying individual trajectories for each joint angle of the robot (or a curvature distribution along the body). While these approaches provide explicit control of the local curvature along the snake body, the macroscopic (i.e. the overall) shape of the robot is only controlled implicitly (i.e. the macroscopic shape is a result of the explicitly controlled curvature along the snake body). The motivation behind this paper is to introduce a control framework which allows the macroscopic shape of a snake robot to be controlled *explicitly* and intuitively.

The control framework is based on specifying the desired shape of the snake robot as a continuous *shape curve* defined by a set of *shape control points* (SCPs). The macroscopic shape of the snake robot is directly controlled by specifying the coordinates and the shape angle associated with each SCP. Note that shape control of snake robots according to continuous curves has been considered in several previous works (see e.g. [2], [3], [8], [10]). However, there are two main contributions of this paper in comparison with previous literature. The first contribution concerns our proposed approach for mathematically describing the desired shape of a snake robot, which is based on interconnecting shape control points by *Bézier curves*. The second contribution is a novel approach for generating gait patterns for a snake robot. The common approach in the literature for shape control based on continuous curves is to continuously deform the curve according to the desired gait. While this approach is also possible with our proposed framework, we also propose to repeatedly *extend* the shape curve according to the gait pattern rather than deforming the shape curve. With this approach, the shape curve grows and 'draws out' the desired shape motion, and joint reference angles corresponding to the gait are retrieved from a *virtual* snake robot which is progressed along the growing shape curve. The motivation behind this approach is that it is (in the authors' opinion) easier and more intuitive to encode a gait pattern by 'drawing' the shape motion as a growing curve rather than encoding the gait in the deformations of an existing curve.

Note that the proposed framework can be employed to specify the SCPs (and thereby the joint reference angles of the snake robot) in open-loop, in which case the framework can be regarded as a *motion planning* framework. However, we still refer to the framework as a *control* framework since practical applications will require the SCPs to be specified as a function of measured state values of the snake robot.

The paper is organized as follows. Section II presents a model of the snake robot dynamics, which has been used to produce the simulation results presented in the paper. Section III provides a short introduction to *Bézier curves*, which have a central role in the control framework presented in Section IV. Practical applications of the proposed control framework are exemplified along with simulation results in Section V. Finally, Section VI presents concluding remarks.

II. A Model of Planar Snake Robot Locomotion

This section summarizes a model of a planar snake robot previously presented in [9]. We will use this model in Section V to simulate various gait patterns implemented with the control framework proposed in Section IV. The model is also presented in order to introduce notation used during the presentation of the control framework.

We consider a planar snake robot consisting of N links of length l interconnected by $N-1$ motorized joints. The kinematics of the robot is defined in terms of the symbols illustrated in Fig. 1. All N links have the same mass m and moment of inertia J. The mass of each link is uniformly distributed so that the link CM (center of mass) is located at its center point. The snake robot moves in the horizontal

Affiliation of Pål Liljebäck is shared between the Dept. of Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway, and SINTEF ICT, Dept. of Applied Cybernetics, N-7465 Trondheim, Norway. E-mail: Pal.Liljeback@sintef.no.

Kristin Y. Pettersen, Øyvind Stavdahl, and Jan Tommy Gravdahl are with the Dept. of Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway. E-mail: {Kristin.Y.Pettersen, Oyvind.Stavdahl, Tommy.Gravdahl}@itk.ntnu.no.

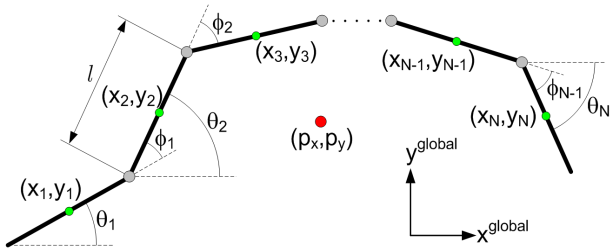


Fig. 1. Kinematic parameters of the snake robot.

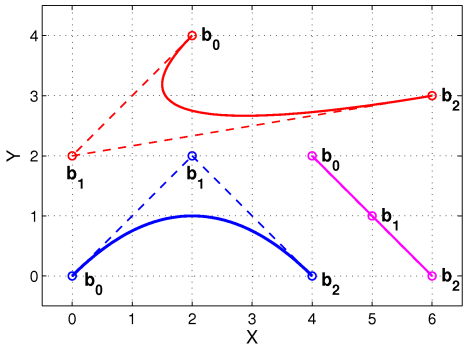


Fig. 2. A plot of three quadratic Bézier curves.

plane and has $N+2$ degrees of freedom. The position of the CM (center of mass) of the robot is denoted by $\boldsymbol{p} = [p_x, p_y]^T \in \mathbb{R}^2$. The absolute angle θ_i of link i is expressed with respect to the global x axis with counterclockwise positive direction. As seen in Fig. 1, the relative angle between link i and link $i+1$ (i.e. the angle of joint i) is defined as $\phi_i = \theta_i - \theta_{i+1}$.

The CM of each link is subjected to a Coulomb ground friction force. The links have anisotropic friction properties, which means that each link has two friction coefficients, μ_t and μ_n, describing the Coulomb friction force in the tangential and normal direction of the link, respectively. For snake robots, it is common to assume that $\mu_n > \mu_t$, which is a property also found in biological snakes [1].

We have shown in [9] that the equations of motion of the snake robot in terms of the joint angles, $\boldsymbol{\phi} \in \mathbb{R}^{N-1}$, the absolute angle of the head link, $\theta_N \in \mathbb{R}$, and the CM position, $\boldsymbol{p}$, can be written as

$$\begin{aligned} &\ddot{\boldsymbol{\phi}} = \boldsymbol{u}, \quad \ddot{\theta}_N = g(\boldsymbol{\phi}, \theta_N, \dot{\boldsymbol{\phi}}, \dot{\theta}_N, \dot{p}_x, \dot{p}_y, \boldsymbol{u}), \\ &Nm\ddot{p}_x = \textstyle\sum_{i=1}^{N} f_{x,i}, \quad Nm\ddot{p}_y = \textstyle\sum_{i=1}^{N} f_{y,i} \end{aligned} \tag{1}$$

where $\boldsymbol{u} \in \mathbb{R}^{N-1}$ is a transformed control input corresponding to the acceleration of the joint angles, $g(\boldsymbol{\phi}, \theta_N, \dot{\boldsymbol{\phi}}, \dot{\theta}_N, \dot{p}_x, \dot{p}_y, \boldsymbol{u}) \in \mathbb{R}$ is a nonlinear function of the state vector and the control input, and where $f_{x,i}$ and $f_{y,i}$ are the Coulomb friction force components on link i in the global x and y direction, respectively.

III. An Introduction to Bézier Curves

In this section, we provide a short introduction to *Bézier curves*, which have a central role in the control framework presented in Section IV. A more detailed description of Bézier curves may be found in e.g. [11], [12].

Bézier curves are parametric curves commonly used in computer graphics to model smooth curves, and are popular due to their compact and intuitive mathematical description. A Bézier curve is defined by a set of *control points* $\boldsymbol{b}_0$, $\boldsymbol{b}_1$, …, $\boldsymbol{b}_n$, where n is the order of the curve. The first ($\boldsymbol{b}_0$) and last ($\boldsymbol{b}_n$) control points are always the end points of the curve. The intermediate control points, however, do not generally lie on the curve. A linear (i.e. first order) Bézier curve is defined in terms of the two control points $\boldsymbol{b}_0$ and $\boldsymbol{b}_1$, and is simply the straight line between these two points, which is written as

$$\boldsymbol{B}(s) = \boldsymbol{b}_0 + s\,(\boldsymbol{b}_1 - \boldsymbol{b}_0) = (1-s)\,\boldsymbol{b}_0 + s\boldsymbol{b}_1 \tag{2}$$

where $s \in [0,1]$ is a curve parameter whose value maps to a location on the line segment between $\boldsymbol{b}_0$ and $\boldsymbol{b}_1$. A quadratic (i.e. second order) Bézier curve is defined in terms of the three control points $\boldsymbol{b}_0$, $\boldsymbol{b}_1$, and $\boldsymbol{b}_2$, and can be regarded as the interpolation between the linear Bézier curves from $\boldsymbol{b}_0$ to $\boldsymbol{b}_1$ and from $\boldsymbol{b}_1$ to $\boldsymbol{b}_2$, respectively. Using (2), this interpolation can be written as

$$\begin{aligned} \boldsymbol{B}(s) = (1-s)\,[(1-s)\,\boldsymbol{b}_0 + s\boldsymbol{b}_1] + \\ s\,[(1-s)\,\boldsymbol{b}_1 + s\boldsymbol{b}_2] \end{aligned} \tag{3}$$

After collecting terms, the quadratic Bézier curve is given as

$$\boldsymbol{B}(s) = (1-s)^2\,\boldsymbol{b}_0 + 2(1-s)s\boldsymbol{b}_1 + s^2\boldsymbol{b}_2 \tag{4}$$

where $s \in [0,1]$. Three examples of quadratic Bézier curves are plotted in Fig. 2. As seen from the plots, the curve tangents at $\boldsymbol{b}_0$ and $\boldsymbol{b}_2$, respectively, both pass through $\boldsymbol{b}_1$. Note that the curve parameter s does not in general correspond to the arc length along the curve. The parameter is always contained in the interval $s \in [0,1]$, where $s = 0$ corresponds to $\boldsymbol{b}_0$ and $s = 1$ corresponds to $\boldsymbol{b}_2$. Higher order Bézier curves can also be defined. In this paper, however, we will only consider quadratic Bézier curves.

IV. A Control Framework for Snake Robot Locomotion

In this section, we propose a control framework for shape control of snake robots. Practical applications of the framework are exemplified in Section V.

A. Overview of the Control Framework

The control framework is based on specifying the desired shape of the snake robot in terms of a continuous curve, denoted the *shape curve*. As illustrated in Fig. 3, the shape curve is a mathematical tool for generating joint reference angles for a snake robot. The shape curve passes through the coordinates of a set of *shape control points* (SCPs). Furthermore, each SCP is associated with an angle which specifies the tangent angle of the shape curve at the location of the SCP. The shape curve is constructed by interconnecting consecutive SCPs by quadratic Bézier curves, which were introduced in Section III. Joint reference angles for the physical snake robot are calculated as the joint angles of a *virtual* snake robot aligned at any desired location along the shape curve. With this approach, the macroscopic shape of the robot is directly controlled with the coordinates and the shape angle associated with each SCP. Note that the SCPs can also be used to explicitly control the local body curvature

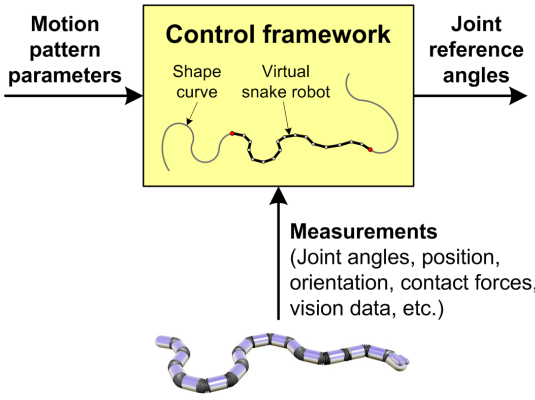


Fig. 3. The general structure of the control framework.

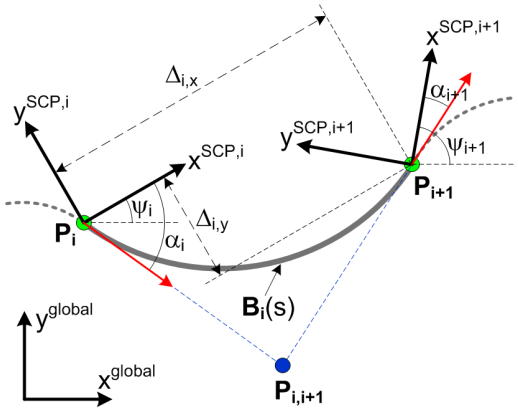


Fig. 4. The parameters defining a shape control point (SCP).

TABLE I
THE PARAMETERS EMPLOYED IN THE CONTROL FRAMEWORK.

Symbol	Description
$\boldsymbol{P}_i = \begin{bmatrix} P_{i,x} \\ P_{i,y} \end{bmatrix}$	The global frame coordinates of SCP i.
$\boldsymbol{\Delta}_i = \begin{bmatrix} \Delta_{i,x} \\ \Delta_{i,y} \end{bmatrix}$	The relative coordinates of SCP $i+1$ with respect to SCP i.
ψ_i	The angle between the local x axis of SCP i and the global x axis.
α_i	The angle of the shape curve tangent at SCP i.
$\boldsymbol{B}_i(s)$	The Bézier curve between SCP i and SCP $i+1$.
$\boldsymbol{S}(s)$	The complete shape curve of the snake robot.
$s_{\text{head}}, s_{\text{tail}}$	The shape curve location of the head and tail of the virtual snake robot.
$s_1, \ldots, s_{N-1}$	The shape curve location of joint $1, \ldots, N-1$ of the virtual snake robot.
$\boldsymbol{\Delta}^{\text{GS}}$	Vector of relative displacements between the SCPs of a gait segment.
$\boldsymbol{\alpha}^{\text{GS}}$	Vector of shape curve angles at each SCP of a gait segment.

along the snake body. The details of the control framework are presented in the following subsections, where we will make use of the parameters summarized in Table I.

B. The Shape Control Point

The *shape control point* (hereafter denoted SCP) represents the building block of the *shape curve* and is defined in terms of the parameters illustrated in Fig. 4. The SCPs are denoted by $\boldsymbol{P}_0, \boldsymbol{P}_1, \ldots, \boldsymbol{P}_{k-1}$, where k is the number of SCPs in the shape curve and $\boldsymbol{P}_i = [P_{i,x}, P_{i,y}]^T \in \mathbb{R}^2$ are the global frame coordinates of the ith SCP. We associate a local coordinate system with each SCP (hereafter denoted the *SCP frame*), where the x and y axes of the ith SCP frame are denoted by $x^{\text{SCP},i}$ and $y^{\text{SCP},i}$, respectively. As seen in Fig. 4, the angle between the $x^{\text{SCP},i}$ axis and the global x axis is denoted by ψ_i with counterclockwise positive direction.

The angle of the shape curve at the ith SCP is denoted by α_i and is described with respect to the local $x^{\text{SCP},i}$ axis. The purpose of describing the shape curve angles with respect to the local SCP frames instead of the global frame is to have more flexibility when designing motion patterns. One useful approach is to always align the x axis of each SCP frame with the desired forward direction of the snake robot. The angle α_i will then specify the angle of the body shape with respect to the forward direction, which is an essential parameter during snake locomotion.

In some situations it may be useful to describe the position of a SCP with respect to the previous SCP instead of the global frame. We therefore define the position of SCP $i+1$ with respect to SCP i by $\boldsymbol{\Delta}_i = [\Delta_{i,x}, \Delta_{i,y}]^T \in \mathbb{R}^2$, where $\boldsymbol{\Delta}_i$ is described in the local coordinate system of SCP i (see Fig. 4). The relation between global frame coordinates and relative coordinates is given by

$$\boldsymbol{P}_{i+1} = \boldsymbol{P}_i + \boldsymbol{R}^{\text{global}}_{\text{SCP},i} \boldsymbol{\Delta}_i \quad , \quad i \in \{0, \ldots, k-2\} \tag{5}$$

where the rotation matrix from the global frame to the ith SCP frame is given by

$$\boldsymbol{R}^{\text{global}}_{\text{SCP},i} = \begin{bmatrix} \cos\psi_i & -\sin\psi_i \\ \sin\psi_i & \cos\psi_i \end{bmatrix} \tag{6}$$

C. The Shape Curve Between Two Shape Control Points

The *shape curve* between each pair of consecutive SCPs is defined as a quadratic Bézier curve (see Section III). Many other types of curves could have been used to interpolate between the SCPs, such as circle segments, elliptical segments, or even straight line segments. However, we chose to employ quadratic Bézier curves due to their compact and analytically tractable form. Moreover, the curve tangents at each end of a Bézier curve are easy to control, which makes it easy to ensure that the shape curve complies with the shape angle specified for each SCP.

As illustrated in Fig. 4, the Bézier curve between SCP i and SCP $i+1$ is denoted by $\boldsymbol{B}_i(s)$. According to Section III, a quadratic Bézier curve is defined in terms of three control points, i.e. two end points and one intermediate point. We choose the coordinates of the two SCPs, i.e. $\boldsymbol{P}_i$ and $\boldsymbol{P}_{i+1}$, as the two end points and we denote the intermediate point by $\boldsymbol{P}_{i,i+1}$, as shown in Fig. 4. The Bézier curve between $\boldsymbol{P}_i$ and $\boldsymbol{P}_{i+1}$ is now given from (4) as

$$\boldsymbol{B}_i(s) = (1-s)^2 \boldsymbol{P}_i + 2(1-s)s\boldsymbol{P}_{i,i+1} + s^2 \boldsymbol{P}_{i+1} \tag{7}$$

where $s \in [0,1]$. The intermediate point $\boldsymbol{P}_{i,i+1}$ can be calculated from the shape curve angles at $\boldsymbol{P}_i$ and $\boldsymbol{P}_{i+1}$,

which are given with respect to the global x axis by $\psi_i + \alpha_i$ and $\psi_{i+1} + \alpha_{i+1}$, respectively. In particular, we know from Section III that $\boldsymbol{P}_{i,i+1}$ is located at the crossing of the two lines which are tangent to the shape curve at $\boldsymbol{P}_i$ and $\boldsymbol{P}_{i+1}$, respectively. These two lines can be written as

$$\boldsymbol{T}_a(s_a) = \boldsymbol{P}_i + s_a \begin{bmatrix} \cos(\psi_i + \alpha_i) \\ \sin(\psi_i + \alpha_i) \end{bmatrix} \tag{8}$$

$$\boldsymbol{T}_b(s_b) = \boldsymbol{P}_{i+1} + s_b \begin{bmatrix} \cos(\psi_{i+1} + \alpha_{i+1}) \\ \sin(\psi_{i+1} + \alpha_{i+1}) \end{bmatrix} \tag{9}$$

where $\boldsymbol{T}_a$ and $\boldsymbol{T}_b$ denote the two line tangents, and $s_a, s_b \in \mathbb{R}$ are the curve parameters along the lines. The crossing of these two line tangents can be found by solving $\boldsymbol{T}_a(s_a) = \boldsymbol{T}_b(s_b)$ for s_a, which gives the solution

$$s_a^* = \frac{P_{i+1,y} - P_{i,y} + \tan(\psi_{i+1} + \alpha_{i+1})(P_{i,x} - P_{i+1,x})}{\sin(\psi_i + \alpha_i) - \cos(\psi_i + \alpha_i)\tan(\psi_{i+1} + \alpha_{i+1})} \tag{10}$$

The intermediate point $\boldsymbol{P}_{i,i+1}$ can therefore be calculated as

$$\boldsymbol{P}_{i,i+1} = \boldsymbol{T}_a(s_a^*) \tag{11}$$

Note that the denominator in (10) is zero when $\psi_i + \alpha_i = \psi_{i+1} + \alpha_{i+1}$, which means that $\boldsymbol{P}_{i,i+1}$ is not properly defined when the line tangents at $\boldsymbol{P}_i$ and $\boldsymbol{P}_{i+1}$ are parallel. A special case occurs when the line tangents are coincident, i.e. when the line tangents are parallel to the straight line from $\boldsymbol{P}_i$ to $\boldsymbol{P}_{i+1}$. In this case, the location of the intermediate point $\boldsymbol{P}_{i,i+1}$ along the two overlapping line tangents is arbitrary, but a natural choice is to place $\boldsymbol{P}_{i,i+1}$ at the center point of the straight line from $\boldsymbol{P}_i$ to $\boldsymbol{P}_{i+1}$.

The singularity occurring when two consecutive shape curve tangents are parallel (except when they are coincident) can easily be avoided by choosing the SCP parameters appropriately. We could also have avoided the singularity by employing cubic (i.e. third order) instead of quadratic Bézier curves to interpolate between the SCPs. However, a cubic Bézier curve has *two* intermediate control points, which means that the coordinates and shape angle at each SCP would not be sufficient to uniquely determine the intermediate points. Moreover, a cubic Bézier curve can always be defined in terms of two quadratic Bézier curves [11], [12], which suggests that the singularity outlined above is caused by a lack of SCPs along the shape curve.

D. The Complete Shape Curve

Given a set of k SCPs, the complete shape curve is spliced together from the $k-1$ Bézier curves $\boldsymbol{B}_i(s)$, $i \in \{0, \ldots, k-2\}$, defined by (7). The complete shape curve is denoted by $\boldsymbol{S}(s)$ and can be compactly written as

$$\boldsymbol{S}(s) = \begin{cases} \boldsymbol{B}_{\lfloor s \rfloor}(s - \lfloor s \rfloor) & , \quad s \in [0, k-1) \\ \boldsymbol{B}_{k-2}(1) & , \quad s = k-1 \end{cases} \tag{12}$$

where the brackets $\lfloor . \rfloor$ mean that the value is rounded down to the nearest integer. To explain the specific form of (12), we provide an example of a shape curve with $k = 5$ SCPs in Fig. 5. Note that the range of the curve parameter s along the complete shape curve $\boldsymbol{S}(s)$ is $s \in [0, k-1]$, while the range of the curve parameter along a single Bézier curve segment $\boldsymbol{B}_i(s)$ is $s \in [0, 1]$. With this convention, the integer part of the curve parameter along $\boldsymbol{S}(s)$ denotes the index i of the current Bézier curve segment $\boldsymbol{B}_i(s)$, while the decimal part denotes the curve parameter along the Bézier curve segment.

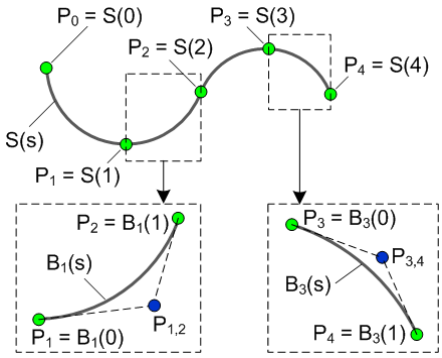


Fig. 5. A shape curve with $k = 5$ SCPs.

E. Generating Joint Reference Angles from the Shape Curve

As described in Section II, the snake robot consists of N links interconnected by $N-1$ motorized joints. We generate joint reference angles for the snake robot by aligning the robot *virtually* at some desired location along the shape curve $\boldsymbol{S}(s)$. To this end, we denote the curve parameters along the shape curve corresponding to the locations of the head and tail of the virtual snake robot by s_{head} and s_{tail}, respectively (see illustration in Fig. 6). Furthermore, we denote the shape curve location of joint i by s_i, where $i \in \{1, \ldots, N-1\}$. The curve parameters $s_{\text{tail}}, s_1, \ldots, s_{N-1}, s_{\text{head}}$ are, in other words, the location of each end point of the N links of the virtual snake robot along the shape curve.

Note that the curve parameters of the virtual snake robot are *not* independent. At any time, it is sufficient to specify only *one* of these parameters since all the remaining parameters will follow from the specific kinematic structure of the snake robot. We have still chosen to introduce explicit notation for the location of each link end point since the point on the snake robot which is employed to specify the location along the shape curve may then be chosen freely.

We have chosen to align the locations of the *joints* to the shape curve since this approach makes the alignment process particularly simple. For example, given a specified curve parameter for the head s_{head}, the location along the shape curve of the joint next to the head s_{N-1} is found by moving along the shape curve (either backward or forward depending on the defined motion planning strategy) until the linear distance to $\boldsymbol{S}(s_{\text{head}})$ equals the link length l. This process must be repeated until all the link ends of the virtual snake robot are aligned, at which point the resulting joint angles of the virtual snake robot represent the joint reference angles for the physical snake robot.

Note that we also could have employed other approaches for aligning the snake robot to the shape curve, such as aligning the center point of each link instead of the link ends. However, when the length l of the links is small, we conjecture that the difference between joint reference angles generated from different alignment methods is negligible.

F. Generating Motion Patterns for the Snake Robot

Motion patterns are defined by the way the shape curve parameters and/or the location of the virtual snake robot along the shape curve are manipulated over time. In the following, we propose two general approaches for generating

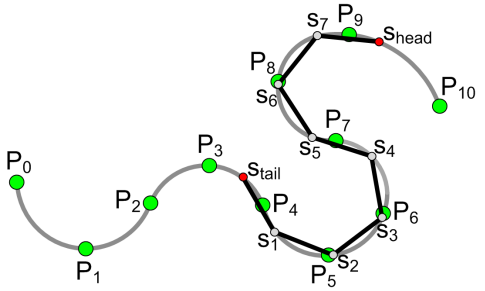


Fig. 6. A *virtual* snake robot with $N = 8$ links aligned along a shape curve with $k = 11$ SCPs.

motion patterns based on the shape curve. The approaches are exemplified in Section V.

1) Virtual snake robot progressing along a growing shape curve: In the first approach, one of the parameters $s_{\text{tail}}, s_1, \ldots, s_{N-1}, s_{\text{head}}$ are explicitly changed at each time step in order to progress the virtual snake robot along the shape curve at some desired velocity $v_{\text{shape}}(t)$ (for example by progressing the head forward along the shape curve such that $\left|\dot{\boldsymbol{S}}(s_{\text{head}})\right| = v_{\text{shape}}(t)$). At each time step, the virtual snake robot is aligned along the shape curve and its joint angles are used as the joint reference angles of the physical snake robot. If, for instance, the shape curve has a sinusoidal shape with several wave cycles, then the physical snake robot will propagate similar sinusoidal wave shapes backwards along its body when the virtual snake robot is progressed forward along the sinusoidal shape curve. Note that the displacement of the virtual snake robot along the shape curve will generally not be equal to the displacement of the physical snake robot. In particular, the displacement of the physical snake robot depends on its ground friction properties, while the displacement of the virtual snake robot is specified directly by the specific motion planning strategy.

When the head or tail of the virtual snake robot reaches the end of the shape curve (for example when s_{head} reaches the value $k-1$), the shape curve should be extended with a new SCP such that the extension of the shape curve reflects the continuation of the desired motion pattern. To facilitate extensions of the shape curve at these discrete time instants, we propose that a *gait segment* is defined for each specific motion pattern. A gait segment defines the form of the shape curve over *one* cycle of a specific motion pattern and is repeatedly concatenated with the shape curve to create a cyclic motion. We denote the number of SCPs in a gait segment by j. Furthermore, since the gait segment is concatenated with the shape curve, we specify the gait segment in terms of the relative coordinates $\boldsymbol{\Delta}$ defined in (5). In particular, we define a general gait segment (GS) by the parameters

$$\boldsymbol{\Delta}^{\text{GS}} = \left[\boldsymbol{\Delta}_0^{\text{GS}}, \boldsymbol{\Delta}_1^{\text{GS}}, \ldots, \boldsymbol{\Delta}_{j-1}^{\text{GS}}\right] \tag{13}$$

$$\boldsymbol{\alpha}^{\text{GS}} = \left[\alpha_0^{\text{GS}}, \alpha_1^{\text{GS}}, \ldots, \alpha_{j-1}^{\text{GS}}\right]^T \tag{14}$$

where $\boldsymbol{\Delta}^{\text{GS}} \in \mathbb{R}^{j\times 2}$ contains the j relative SCP coordinates of the gait segment and $\boldsymbol{\alpha}^{\text{GS}} \in \mathbb{R}^j$ contains the shape curve angle of each SCP. Practical use of the gait segment is exemplified in Section V.

2) Virtual snake robot fixed along a continuously deformed shape curve: The second approach for generating motion patterns from the shape curve is to fix the parameters $s_{\text{tail}}, s_1, \ldots, s_{N-1}, s_{\text{head}}$ and instead modify the coordinates and/or shape angle of the existing SCPs according to the desired motion pattern. At each time step, the virtual snake robot is aligned along the shape curve, which means that the generated joint reference angles reflect the deformations of the shape curve. In the authors' opinion, this second approach is more complex than the first approach described above since it is easier and more intuitive to encode a gait pattern by 'drawing' the shape motion as a growing curve rather than encoding it in the deformations of an existing curve.

V. Examples of Applications of the Control Framework

In this section, we demonstrate how the proposed control framework can be employed in practice to control snake robot locomotion. The section presents simulation results based on the model of the snake robot in (1).

A. Implementation of the Model and the Control Framework

The model of the snake robot in (1) and the control framework presented in Section IV were implemented in *Matlab R2008b*. The dynamics of the model was calculated using the *ode3* solver with a step length of 0.01 s.

We considered a snake robot with links of length $l = 0.1$ m, mass $m = 1$ kg, and moment of inertia $J = ml^2/3$. The number of links N and the friction coefficients of the links, μ_t and μ_n, are presented with each simulation result below. All initial state values of the robot were zero.

In order to make the joint angles $\boldsymbol{\phi} = [\phi_1, \ldots, \phi_{N-1}]^T \in \mathbb{R}^{N-1}$ of the simulated snake robot track the joint reference angles $\boldsymbol{\phi}_{\text{ref}} = [\phi_{1,\text{ref}}, \ldots, \phi_{N-1,\text{ref}}]^T \in \mathbb{R}^{N-1}$ from the virtual snake robot on the shape curve, we set the control input $\boldsymbol{u} \in \mathbb{R}^{N-1}$ of the model (1) according to the controller

$$\boldsymbol{u} = k_p\left(\boldsymbol{\phi}_{\text{ref}} - \boldsymbol{\phi}\right) - k_d\dot{\boldsymbol{\phi}} \tag{15}$$

where $k_p > 0$ and $k_d > 0$ are controller gains, and where $\dot{\boldsymbol{\phi}}_{\text{ref}} = \boldsymbol{0}$ since the purpose of the derivative part is simply to dampen the joint motion if the joint velocities become large.

B. Lateral Undulation

Lateral undulation is the most common form of snake locomotion and is achieved by propagating continuous waves backwards along the snake body from head to tail [1]. During this wave motion, the sides of the snake body push against irregularities in the surface, thereby pushing the snake forward. To demonstrate how the proposed control framework can be used to generate joint reference angles for this gait pattern, we considered a snake robot with $N = 10$ links with friction coefficients $\mu_t = 0.1$ and $\mu_n = 1$. The shape curve was initialised as a single SCP with parameters $\boldsymbol{P}_0 = [0, 0]^T$, $\alpha_0 = 0°$, and $\psi_0 = 0°$. Furthermore, we employed the approach described in Section IV-F.1 and specified the gait segment parameters in (13) and (14) as

$$\boldsymbol{\Delta}^{\text{GS}} = l\begin{bmatrix} 3/2 & 3/2 & 3/2 & 3/2 \\ 1 & 1 & -1 & -1 \end{bmatrix} \tag{16}$$

$$\boldsymbol{\alpha}^{\text{GS}} = \begin{bmatrix} 60° & 0° & -60° & 0° \end{bmatrix}^T \tag{17}$$

Using (5), the gait segment is plotted in Fig. 7(a) with respect to the initial shape curve. The figure shows that the Bézier curve interpolation in (7) between each SCP creates a smooth curve which complies with the shape angles specified in (17).

To demonstrate how the control framework can be employed to achieve directional control of the locomotion, we set the frame orientation ψ_i of each new SCP according to the directional controller

$$\psi_i = k_\theta \left(\overline{\theta}_{\text{ref}} - \overline{\theta}\right) \tag{18}$$

where i is the index of the SCP, $k_\theta > 0$ is a controller gain, $\overline{\theta}_{\text{ref}}$ is the reference direction of the motion, and $\overline{\theta}$ is the current orientation of the simulated snake robot, which was estimated as the average of the link angles, i.e. as $\overline{\theta} = 1/N \sum_{i=1}^{N} \theta_i$. For each SCP added to the shape curve, the controller in (18) rotates the SCP frame in the direction towards which the robot should turn. This affects the direction of motion since the gait segment is defined with respect to the SCP frames. Note that the orientation of each SCP frame was fixed after being set at the time instant when the SCP was added to the shape curve. We chose the reference direction as $\overline{\theta}_{\text{ref}} = 0^\circ$ for $t \in [0, 30)$ and $\overline{\theta}_{\text{ref}} = 45^\circ$ for $t \in [30, 70]$. The controller gain was $k_\theta = 0.2$.

To generate joint reference angles, we initialised the location of the head of the virtual snake robot along the shape curve as $s_{\text{head}} = 0$ and propagated the virtual snake forward such that $\left|\dot{\boldsymbol{S}}(s_{\text{head}})\right| = v_{\text{shape}}(t) = 15$ cm/s. The SCPs in (16) were consecutively added (one by one) to the shape curve each time s_{head} reached the end of the shape curve ($s_{\text{head}} = k-1$). When the last SCP of (16) was added, the concatenation process started over from the first SCP of (16) in order to create a cyclic motion pattern. Note that the virtual snake robot was initially located *outside* the shape curve (except for the head). By convention, the angles of joints outside the shape curve were set to zero.

The resulting shape curve after $t = 70$ s of simulated motion is plotted in Fig. 7(b), where the shape curve location of the virtual snake robot is plotted for $t = 10$ s, $t = 30$ s, and $t = 65$ s, respectively. A close-up of the virtual snake robot at $t = 30$ s is shown in Fig. 7(c). The resulting motion of the simulated snake robot is shown in Fig. 7(d), where we see that the locomotion is parallel with the global x axis at first (in accordance with $\overline{\theta}_{\text{ref}} = 0^\circ$), but changes according to $\overline{\theta}_{\text{ref}} = 45^\circ$ after $t = 30$ s. Note that the robot turned slightly to its right during the first seconds of the simulation since the directional controller did not have full effect until the virtual snake robot had entered the shape curve completely.

As explained in Section IV-F.1 and clearly illustrated in Figures 7(b) and (d), the displacement of the virtual and the simulated snake robot, respectively, are generally not equal. In particular, the virtual snake robot follows the shape curve, which expands in the direction defined by the controller in (18). The displacement of the simulated snake robot, on the other hand, depends on its ground friction properties.

Remark 1: The common approach in the literature for achieving lateral undulation is to control each joint angle according to a sinusoidal reference trajectory [9]. With this approach, however, it is not trivial to determine the resulting macroscopic shape of the body wave motion, which is a geometric function of the robot kinematics and the sinusoidal joint motion. The simulation results in Fig. 7 illustrate that the control framework allows the macroscopic shape motion to be controlled explicitly and intuitively.

Remark 2: The common approach in the literature for achieving directional control during lateral undulation is to add an offset to the sinusoidal reference trajectory of each joint [9]. This approach causes the entire snake to attempt to turn simultaneously. On the other hand, biological snakes turn during lateral undulation by first introducing a turning action at the head, which is repeated in turn by the consecutive body segments. This approach, which we believe to be more efficient than turning with the whole snake body at once, is achieved with the directional controller in (18) since a turning action is propagated backwards along the snake body according to the forward progression of the virtual snake robot along the shape curve.

C. Concertina Motion

A snake robot can display a concertina-like motion pattern by curling (i.e. contracting) a small part of its body while the rest of the body is lying straight, and then propagating the curled shape forward along the body. When the curled shape has propagated over the entire body length, the robot will have displaced its center of mass a step forward. We implemented this motion pattern using the exact same approach as in Section V-B. In particular, we defined the gait segment parameters in (13) and (14) as

$$\boldsymbol{\Delta}^{\text{GS}} = l \begin{bmatrix} 0 & 1 & 1 & 1 & l & 1 & 1 & 10 \\ 0 & 3/2 & -3/2 & -3/2 & 3/2 & 3/2 & -3/2 & 0 \end{bmatrix} \tag{19}$$

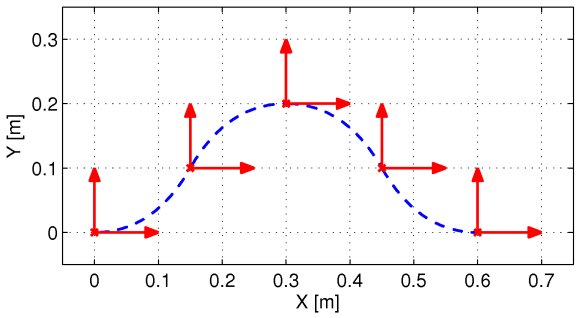
$$\boldsymbol{\alpha}^{\text{GS}} = \begin{bmatrix} 80^\circ & 0^\circ & -80^\circ & 0^\circ & 80^\circ & 0^\circ & -80^\circ & 0^\circ \end{bmatrix}^T \tag{20}$$

The gait segment is plotted in Fig. 8(a) and consists of an initial curl followed by a long straight segment. We considered a snake robot with $N = 20$ links and friction coefficients $\mu_t = \mu_n = 0.1$, which were equal since concertina motion does not require anisotropic ground friction. The shape curve propagation velocity was $v_{\text{shape}}(t) = 20$ cm/s.

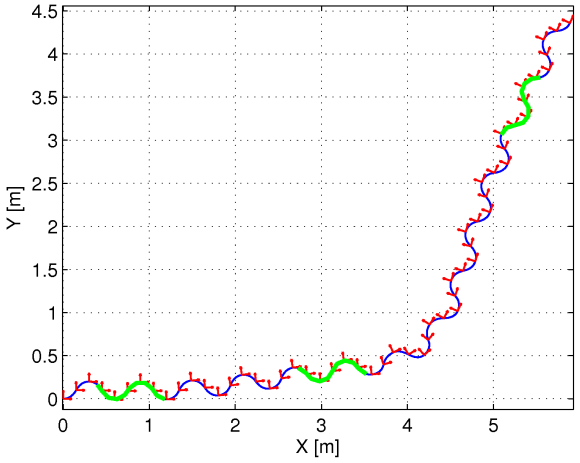
The simulated motion of the snake robot is shown in Fig. 8(b). Note that the robot moved in the negative global x direction since the curled shape travelled backwards along the body. Although the motion was not very efficient, it is clearly seen that the motion pattern successfully propelled the robot. This simulation result illustrates how a completely different gait pattern easily can be designed within the control framework simply by redefining the gait segment.

D. Lateral Rolling

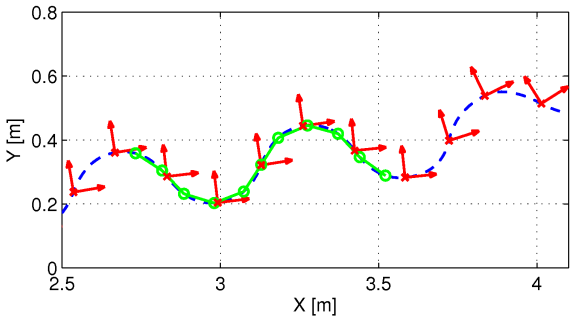
During lateral rolling, a snake robot undergoes a continuous rolling motion about its tangential body axis (see e.g. [9]), which requires that the snake robot can form its body into an arc in both the horizontal and the vertical plane. In the following, we will demonstrate how the control framework can be employed to generate a 'planar version' of lateral rolling, where the body of the snake is continuously bent back and forth in the horizontal plane. Although this motion will not produce propulsion, the simulation results demonstrate that lateral rolling easily can be achieved with the proposed control framework by extending the framework with 3D curves (which is a topic of future work).



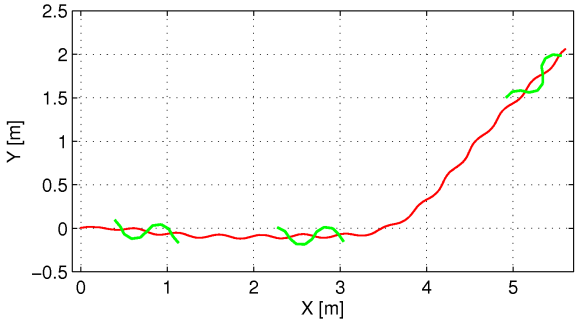
(a) The gait segment of lateral undulation.



(b) The shape curve after $t = 70$ s.

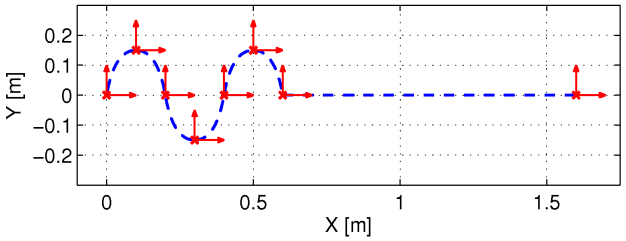


(c) Close-up of the virtual snake robot at $t = 30$ s.

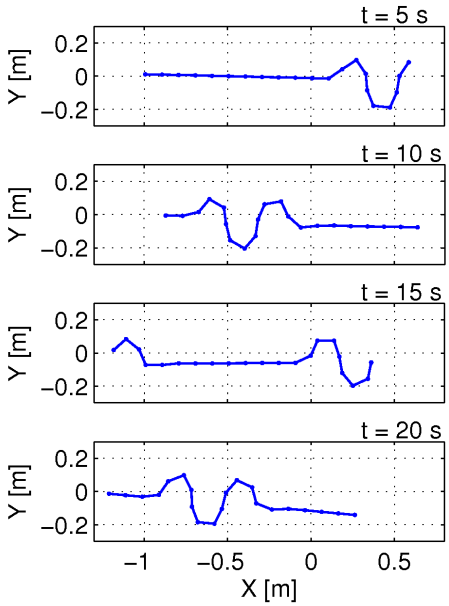


(d) The resulting motion of the snake robot.

Fig. 7. Simulation of lateral undulation. The reference direction is $\overline{\theta}_{\mathrm{ref}} = 0^\circ$ for $t < 30$ s and $\overline{\theta}_{\mathrm{ref}} = 45^\circ$ for $t \geq 30$ s.



(a) The gait segment of concertina-like motion.



(b) The resulting motion of the simulated snake robot.

Fig. 8. Simulation of a concertina-like motion pattern.

Since the approach of progressing the virtual snake robot forward along the shape curve is not suitable for producing a continuous bending motion back and forth, we chose to employ the approach described in Section IV-F.2. In particular, we defined the shape curve in terms of the following three SCPs with time-varying coordinates:

$$\boldsymbol{P}_0 = \begin{bmatrix} 0 \\ 2l\sin\left(\frac{\pi}{4}t\right) \end{bmatrix}, \boldsymbol{P}_1 = \begin{bmatrix} Nl/2 \\ 0 \end{bmatrix}, \boldsymbol{P}_2 = \begin{bmatrix} Nl \\ 2l\sin\left(\frac{\pi}{4}t\right) \end{bmatrix} \quad (21)$$

Furthermore, we defined the shape curve angles as

$$\begin{aligned} &\alpha_0 = -\tfrac{\pi}{3}\sin\left(\tfrac{\pi}{4}t\right) \quad , \quad \alpha_1 = 0 \quad , \quad \alpha_2 = \tfrac{\pi}{3}\sin\left(\tfrac{\pi}{4}t\right) \\ &\psi_0 = 0 \qquad\qquad\quad , \quad \psi_1 = 0 \quad , \quad \psi_2 = 0 \end{aligned} \quad (22)$$

This time-varying shape curve is plotted to the left in Fig. 9, where we can see that the shape curve is bent back and forth in the horizontal plane. In order to produce joint reference angles corresponding to this shape motion for a snake robot with $N = 10$ links, we simply fixed the middle joint of the virtual snake robot in the second SCP by setting $s_5 = 1$. The simulated motion of the snake robot according to the model in (1) is shown in Fig. 9. As expected, the snake robot displayed the desired arc shape motion.

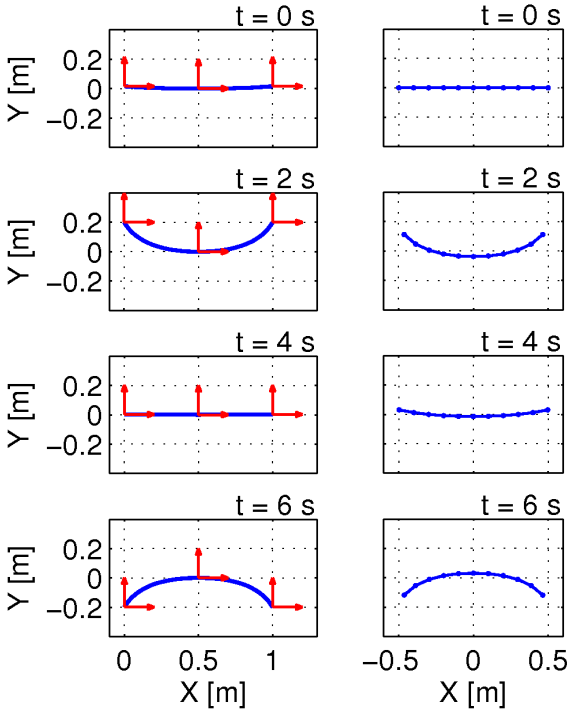


Fig. 9. The planar motion component of *lateral rolling*. Left: A shape curve with three SCPs, where the first and third SCP oscillate about the global x axis. Right: The resulting motion of the snake robot.

Notice how the control framework allows us to easily and explicitly specify the desired time-varying shape of the snake robot without being concerned about the motion of the individual joints. Note also that if this approach is employed in a 3D setting in order to achieve actual rolling motion, then directional control can be achieved by displacing the SCPs at each end point with different amplitudes.

E. Achieving Environment Adaptation

The gait patterns demonstrated previously in this section work well for snake robot locomotion over relatively flat surfaces. However, efficient locomotion in cluttered and challenging environments, which is more in line with practical applications of snake robots, requires that the robot can *sense* its environment and *adapt* its body shape and movements accordingly [9]. We argue that the proposed control framework is particularly well suited for achieving environment adaptation during locomotion. In particular, since the framework allows us to explicitly specify the macroscopic shape of a snake robot in terms of shape control points, we can achieve environment adaptation by placing shape control points in direct accordance with the geometry of the environment (which can be estimated from e.g. contact force measurements). This approach is illustrated in Fig. 10. Investigating the applicability of the control framework for motion in cluttered environments represents an important topic of future work.

VI. Conclusions

This paper has presented a control framework for snake robot locomotion which allows the macroscopic shape of a snake robot to be controlled explicitly. The framework is based on specifying the desired shape of the snake robot as a continuous *shape curve* defined by a set of *shape control points* interconnected by *Bézier curves*. We have proposed an approach for motion generation in which the shape curve is repeatedly extended according to a desired gait pattern while a virtual snake robot is progressed along the shape curve to retrieve joint reference angles. Practical applications of the proposed control framework were exemplified along with simulation results. In future work, the authors will extend the framework to support 3D curves.

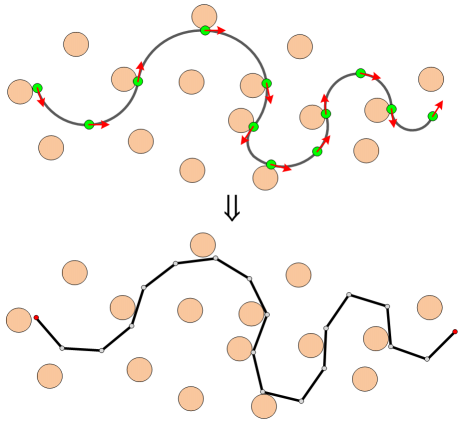


Fig. 10. Top: A shape curve defined in terms of shape control points placed with respect to detected external objects. Bottom: A snake robot adapting to its environment by tracking the shape of the shape curve.

References

[1] S. Hirose, *Biologically Inspired Robots: Snake-Like Locomotors and Manipulators*. Oxford: Oxford University Press, 1993.

[2] G. Chirikjian and J. Burdick, "The kinematics of hyper-redundant robot locomotion," *IEEE Trans. Robot. Autom.*, vol. 11, no. 6, pp. 781–793, December 1995.

[3] H. Yamada and S. Hirose, "Study on the 3d shape of active cord mechanism," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2006, pp. 2890–2895.

[4] K. Lipkin, I. Brown, H. Choset, J. Rembisz, P. Gianfortoni, and A. Naaktgeboren, "Differentiable and piecewise differentiable gaits for snake robots," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, San Diego, CA, USA, Oct-Nov 2007, pp. 1864–1869.

[5] R. Hatton and H. Choset, "Generating gaits for snake robots by annealed chain fitting and keyframe wave extraction," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2009, pp. 840–845.

[6] P. Prautsch, T. Mita, and T. Iwasaki, "Analysis and control of a gait of snake robot," *Trans. Institute of Electrical Engineers of Japan*, vol. 120-D, no. 3, pp. 372–381, 2000.

[7] S. Ma, "Analysis of creeping locomotion of a snake-like robot," *Adv. Robotics*, vol. 15, no. 2, pp. 205–224, 2001.

[8] H. Date and Y. Takita, "Control of 3d snake-like locomotive mechanism based on continuum modeling," in *Proc. ASME 2005 International Design Engineering Technical Conferences*, 2005, pp. 1351–1359.

[9] P. Liljebäck, K. Y. Pettersen, Ø. Stavdahl, and J. T. Gravdahl, *Snake Robots - Modelling, Mechatronics, and Control*, ser. Advances in Industrial Control. Springer, 2012.

[10] Z. Y. Bayraktaroglu, "Snake-like locomotion: Experimentations with a biologically inspired wheel-less snake robot," *Mechanism and Machine Theory*, vol. 44, no. 3, pp. 591–602, 2008.

[11] G. Farin, *Curves and Surfaces for CAGD: A Practical Guide*, 5th ed., ser. The Morgan Kaufmann Series in Computer Graphics. Morgan Kaufmann, 2001.

[12] D. Salomon, *Curves and Surfaces for Computer Graphics*, 1st ed. Springer, 2005.