A 3D Motion Planning Framework for Snake Robots

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

Abstract— **This paper presents a motion planning framework for three-dimensional body shape control of snake robots. Whereas conventional motion planning approaches define the body shape of snake robots in terms of their individual joint angles, the proposed framework allows the body shape to be specified in terms of Cartesian coordinates in the environment of the robot. This approach simplifies motion planning since Cartesian coordinates are more intuitively mapped to the overall body shape of the snake robot. The paper demonstrates the applicability of the framework for realizing different types of three-dimensional motion patterns.**

I. Introduction

Snake robots are robotic mechanisms designed to move like biological snakes [1]. The advantage of such mechanisms is their long and flexible body, which gives them the potential to move and operate in challenging environments where human presence is undesirable or impossible. Potential applications of these mechanisms include search and rescue operations, inspection and maintenance, and subsea operations. Due to their complex dynamics and unique forms of propulsion, snake robots pose many open research problems.

Numerous approaches to motion control of snake robots have been proposed in the literature, and the reader is referred to [2] for a detailed overview. The majority of previous approaches specify *explicitly* the motion of each individual joint angle of the snake robot. Motion planning on the joint angle level is, however, a particularly challenging task if we are primarily interested in controlling the overall (macroscopic) body shape of the robot, which is generally the case. For this reason, it would simplify the control problem if we could specify the motion in terms of parameters which are more intuitively mapped to the overall body shape.

This idea is related to *joint space* versus *task space* control of a conventional robot manipulator [3], whose motion is usually specified in the task space of the end effector and then mapped to the motion of each joint. The 'task' space of a snake robot is, however, defined in terms of the position of all the links (i.e. not only the end effector) since it is the interaction between the links and the physical environment that creates the propulsion forces. Moreover, unlike most robot manipulators, snake robots are underactuated since they have more degrees of freedom than independent control inputs. For these reasons, motion planning methods for robot manipulators are not directly applicable to snake robots.

Motivated by the above discussion, we propose in this paper a general framework which facilitates three-dimensional motion planning for snake robots. Instead of specifying joint angles, the framework allows the motion to be specified in terms of coordinates of *shape control points* (SCPs). A continuous curve between the SCPs defines the desired shape of the snake robot and is mapped to the individual joint angles through a curve fitting process. The paper demonstrates how the framework can be applied to design complex three-dimensional motion patterns for snake robots.

Motion planning for snake robots based on continuous curves has also been considered in previous literature [4]–[7]. This paper is, however, different from these previous works both in terms of the parametrization of the continuous curve and in terms of the approach for generating dynamic motion patterns based on the continuous curve.

This paper builds on results presented by the authors in [8], where this motion planning framework was considered in the context of planar (two-dimensional) motion. This paper extends the previous results by showing how the framework can be extended to specify three-dimensional motion.

The paper is organized as follows. Section II presents parameters of the snake robots considered in the paper. The motion planning framework is presented in Section III, and its applicability for three-dimensional motion planning is demonstrated by simulation results presented in Section IV. Finally, Section V presents concluding remarks.

II. The Snake Robot

This section presents the kinematic structure of the class of snake robots considered in this paper. Properties related to the dynamics of the robot (mass, max actuator torque, etc.) are not specified since the proposed motion planning approach is carried out on a purely kinematic level.

The general kinematics of the snake robot is illustrated in Fig. 1 using the parameters summarized in Table I. In particular, the snake robot consists of N serially connected rotational joints, where the rotation axes of consecutive joints are orthogonal. The majority of existing snake robots belongs in this class, including snake robots with 2-DOF joint modules where the yaw and pitch axes are intersecting.

The kinematic parameters illustrated in Fig. 1 have been defined according to the *Denavit-Hartenberg Convention* [3], which is a systematic procedure for modelling the kinematics of serially connected joint mechanisms. The specific Denavit-Hartenberg (DH) parameters of the snake robot are listed in Table II. In particular, a_i defines the link length between joint i and joint $i+1$, d_i defines the transversal offset of joint $i+1$ with respect to joint i (which is assumed to be zero), α_i defines the relative angle between the rotation axis of joint i and joint $i+1$, and θ_i defines the angle between the

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

Kristin Y. Pettersen is with the Centre for Autonomous Marine Operations and Systems, Dept. of Engineering Cybernetics at NTNU, 7491 Trondheim, Norway. E-mail: Kristin.Y.Pettersen@itk.ntnu.no.

Øyvind Stavdahl and Jan Tommy Gravdahl are with the Dept. of Engineering Cybernetics at NTNU, 7491 Trondheim, Norway. E-mail: {Oyvind.Stavdahl, Tommy.Gravdahl}@itk.ntnu.no.

This work was partly supported by the Research Council of Norway through project no. 205622 and its Centres of Excellence funding scheme project no. 223254.

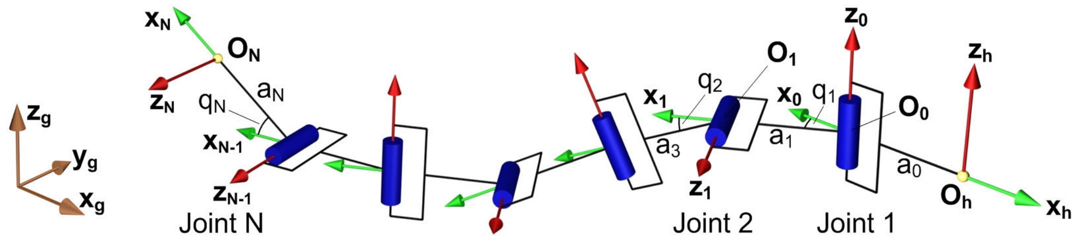


Fig. 1. The kinematics of the snake robot modelled according to the Denavit-Hartenberg Convention. The y axis of each frame is not shown.

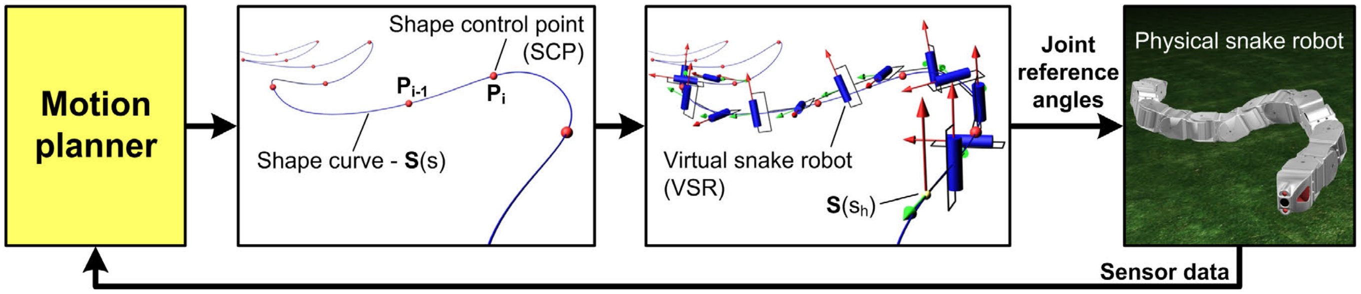


Fig. 2. Overview of the motion planning framework.

x_{i-1} and x_i axis. The angle of joint i is denoted by q_i and corresponds to the parameter θ_i for $i \in \{1, \dots, N\}$. Note that the parameter α_{N-1} is $-\frac{\pi}{2}$ if N is an even number and $\frac{\pi}{2}$ if N is an odd number. The reader is referred to e.g. [3] for a more detailed description of these parameters.

The global frame position and orientation of the robot's head tip is defined by the homogenous transformation matrix

$$\boldsymbol{T}_h^g = \begin{bmatrix} \boldsymbol{x}_h & \boldsymbol{y}_h & \boldsymbol{z}_h & \boldsymbol{O}_h \\ 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \tag{1}$$

Furthermore, the global frame position and orientation of frame $i \in \{0, \dots, N\}$ is calculated as

$$\boldsymbol{T}_i^g = \begin{bmatrix} \boldsymbol{x}_i & \boldsymbol{y}_i & \boldsymbol{z}_i & \boldsymbol{O}_i \\ 0 & 0 & 0 & 1 \end{bmatrix} = \boldsymbol{T}_h^g \boldsymbol{T}_0^h \boldsymbol{T}_1^0(q_1) \boldsymbol{T}_2^1(q_2) \cdots \boldsymbol{T}_i^{i-1}(q_i) \tag{2}$$

where

$$\boldsymbol{T}_0^h = \begin{bmatrix} -1 & 0 & 0 & -a_0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

and where $\boldsymbol{T}_i^{i-1}(q_i)$ is defined for $i \in \{1, \dots, N\}$ as

$$\boldsymbol{T}_i^{i-1}(q_i) = \begin{bmatrix} \cos q_i & -\sin q_i \cos\alpha_i & \sin q_i \sin\alpha_i & a_i \cos q_i \\ \sin q_i & \cos q_i \cos\alpha_i & -\cos q_i \sin\alpha_i & a_i \sin q_i \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

III. The Motion Planning Framework

In this section, we propose a general approach for specifying the three-dimensional motion of snake robots.

A. Motivation and Overview of the Framework

The propulsion of snake robots is generated by body shape changes which induce contact forces from the environment that propel the robot [2]. The most fundamental objective of a control strategy for a snake robot is therefore to control its body shape. Previous control approaches for snake robots solve this problem by specifying directly the motion of each individual joint. A challenge with this direct joint control approach is that the mapping from individual joint angles to the overall body shape of a snake robot is generally quite complex. For this reason, it would simplify the control problem if we could specify the motion in terms of parameters which are more intuitively mapped to the overall body shape of the snake robot. In particular, a tool which simplifies body shape control will also simplify the task of adapting the motion in challenging and cluttered environments.

The motion planning framework proposed in the following is motivated by the above discussion and is summarized in Fig. 2. In particular, the desired shape of the snake robot is specified in terms of *shape control points* (hereafter denoted SCPs). The SCPs are interconnected by a curve denoted the *shape curve*, which defines the macroscopic shape of the snake robot. A *virtual snake robot* (hereafter denoted VSR)

TABLE I
The kinematic parameters of the snake robot.

Symbol	Description
N	Number of joints.
$\boldsymbol{x}_g,\boldsymbol{y}_g,\boldsymbol{z}_g \in \mathbb{R}^3$	Coordinate axes of global frame.
$\boldsymbol{x}_h,\boldsymbol{y}_h,\boldsymbol{z}_h \in \mathbb{R}^3$	Coordinate axes of head frame.
$\boldsymbol{x}_i,\boldsymbol{y}_i,\boldsymbol{z}_i \in \mathbb{R}^3$	Coordinate axes of frame $i \in \{0, \dots, N\}$. The rotation axis of joint $i \in \{1, \dots, N\}$ is $\boldsymbol{z}_{i-1}$.
$\boldsymbol{O}_h \in \mathbb{R}^3$	Origin of frame attached to the robot's head tip.
$\boldsymbol{O}_i \in \mathbb{R}^3$	Origin of frame $i \in \{0, \dots, N\}$.
a_0,$a_N \in \mathbb{R}$	Length of head and tail link, respectively.
$a_i \in \mathbb{R}$	Link length between joint i and joint $i+1$.
$q_i \in \mathbb{R}$	Angle of joint $i \in \{1, \dots, N\}$.

TABLE II
The Denavit-Hartenberg parameters of the snake robot.

$\boldsymbol{i}$	$\boldsymbol{a}_i$	$\boldsymbol{d}_i$	$\boldsymbol{\alpha}_i$	$\boldsymbol{\theta}_i$
0	a_0	0	0	π
1	a_1	0	$-\frac{\pi}{2}$	q_1
2	a_2	0	$\frac{\pi}{2}$	q_2
3	a_3	0	$-\frac{\pi}{2}$	q_3
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$N-1$	a_{N-1}	0	$-\frac{\pi}{2}$	q_{N-1}
N	a_N	0	0	q_N

TABLE III
The parameters of the motion planning framework.

Symbol	**Description**
n	Number of SCPs in the shape curve.
$\boldsymbol{P}_i \in \mathbb{R}^3$	Coordinates of the ith SCP.
$\boldsymbol{S}(s) \in \mathbb{R}^3$	Shape curve which interconnects the SCPs.
$s_h \in \mathbb{R}$	Shape curve location of the VSR's head.
$\phi_h \in \mathbb{R}$	Roll angle of VSR about the shape curve.
$s_{\text{ref}} \in \mathbb{R}$	Reference point on the shape curve when aligning the links of the VSR.
$l_{\text{LAD}} \in \mathbb{R}$	The look-ahead distance of the alignment algorithm.

with identical kinematic structure as the physical robot is aligned along the shape curve, starting from some specified location along the curve. The joint reference angles for the physical robot are then defined as the joint angles of the VSR. Furthermore, dynamic motion patterns for the physical robot are produced by varying the SCP coordinates and/or the location of the VSR along the shape curve with time.

As illustrated in Fig. 2, motion planning is carried out within this framework by specifying the Cartesian coordinates of the SCPs and the location and orientation of the VSR's head tip along the shape curve. The joint reference angles for the physical snake robot follow implicitly from the explicitly specified body shape since the framework automatically aligns the VSR along the shape curve.

The details of this motion planning approach are presented in the following subsections, where we will make use of the parameters listed in Table III.

B. The Shape Control Points and the Shape Curve

To construct the shape curve defining the desired shape of the snake robot, we begin by defining a set of SCPs. The global frame coordinates of the SCPs are denoted by $\boldsymbol{P}_0, \boldsymbol{P}_1, \ldots, \boldsymbol{P}_{n-1}$, where n is the number of SCPs in the shape curve and $\boldsymbol{P}_i = [P_{i,x}, P_{i,y}, P_{i,z}]^T \in \mathbb{R}^3$. The number n of SCPs is generally not fixed since, as exemplified later in this section, the motion of the snake robot can be defined by periodically extending the shape curve with SCPs according to the desired motion pattern.

The shape curve interconnecting the SCPs is denoted by $\boldsymbol{S}(s)$, where $s \in [0, n-1]$ is the scalar shape curve parameter. The curve is produced by interpolating between the SCPs using any chosen interpolation method such that $\boldsymbol{S}(i) = \boldsymbol{P}_i$, where $i \in \{0, \ldots, n-1\}$. Note that motion planning strategies for the SCPs can be specified independently from the choice of interpolation.

Example 1: A shape curve based on $n = 4$ SCPs is plotted in Fig. 4(a). The SCP coordinates are $\boldsymbol{P}_0 = (0, 0, 0)$, $\boldsymbol{P}_1 = (0.25, 0.15, 0)$, $\boldsymbol{P}_2 = (0.5, 0, 0.05)$, and $\boldsymbol{P}_3 = (0.75, -0.15, 0.3)$. The shape curve $\boldsymbol{S}(s)$ was constructed using a *piecewise cubic hermite interpolating polynomial* [9], which is produced in *Matlab* using the function '*pchip*'. This interpolation method creates smooth curves which do not overshoot the SCPs. The coordinates of any point along the shape curve is retrieved using the shape curve parameter $s \in [0, 3]$. For instance, we have that $\boldsymbol{S}(0) = \boldsymbol{P}_0$, $\boldsymbol{S}(2) = \boldsymbol{P}_2$, and $\boldsymbol{S}(2.5)$ gives the coordinates of the point midway between $\boldsymbol{P}_2$ and $\boldsymbol{P}_3$ on the shape curve.

C. The Virtual Snake Robot (VSR)

The shape curve $\boldsymbol{S}(s)$ defines the shape that we want the snake robot to attain. As illustrated in Fig. 2, this curve is mapped to joint reference angles for the physical snake robot by aligning a VSR (with identical kinematic structure as the physical robot) along the curve and then using the resulting joint angles of the VSR as reference angles for the physical robot. The motion planner only needs to specify the location of the head tip of the VSR on the shape curve. The alignment of the VSR is subsequently carried out automatically within the framework, which means that a motion pattern can be designed on the shape curve level without concern about the motion of the joints. As explained in the next subsection and illustrated in Fig. 3(a), we define the location of the VSR's head tip in terms of the shape curve parameter $s_h \in [0, n-1]$ and the roll angle ϕ_h about the shape curve.

Remark 2: The shape curve does not necessarily need to be produced from the SCPs before specifying the location s_h of the VSR's head tip since, by design, we know that $s_h \in [0, n-1]$ and that $\boldsymbol{S}(i) = \boldsymbol{P}_i$ for $i \in \{0, \ldots, n-1\}$ regardless of how the shape curve is produced from the SCPs.

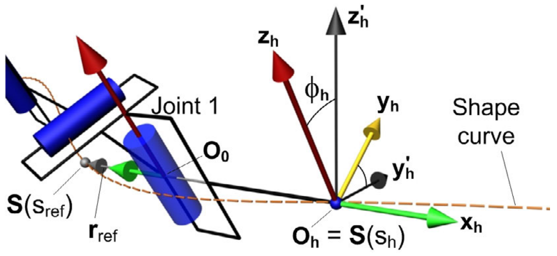
D. The Alignment of the VSR along the Shape Curve

Since the snake robot consists of joints interconnected by straight links, it is generally not possible to align the VSR perfectly along the continuous shape curve. There are, however, many ways of achieving an *approximate* fit of the VSR along the curve. The alignment strategy that we propose in the following is not optimal by any specific measure, but performs well in terms of fitting the VSR along general three-dimensional shape curves. Moreover, a great advantage of the algorithm is its computational simplicity, which facilitates motion planning in real-time. Note that motion planning on the shape curve level can be carried out independently of the chosen method for aligning the VSR along the curve.

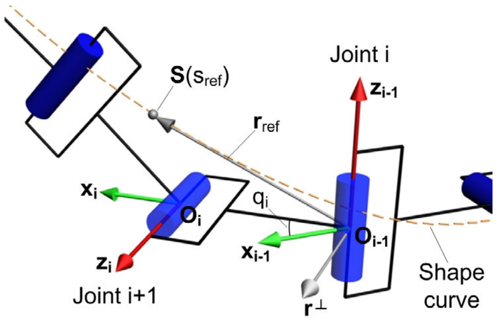
A detailed description of the algorithm is given in the following subsections. The algorithm is summarized in Fig. 3 and aligns the VSR starting with the head tip at the desired location and then working its way backwards to the tail.

1) The Input to the Algorithm: The alignment algorithm takes two scalar inputs. The first input is the shape curve parameter $s_h \in [0, n-1]$ of the head tip of the VSR and the second input is its roll angle ϕ_h about the shape curve.

2) The Output from the Algorithm: As illustrated in Fig. 1 and described in Table I, the kinematics of the snake robot is defined in terms of $N+2$ coordinate systems, i.e. the h frame



(a) Alignment of the head frame.



(b) Alignment of frame $i \in \{0, \dots, N\}$.

Fig. 3. The strategy for aligning the VSR along the shape curve. (a) The VSR is aligned from the head link and backwards. The location s_h of the head tip and the roll angle ϕ_h of the VSR about the shape curve are inputs to the algorithm. (b) The link from frame $i-1$ to frame i is 'aimed' towards a reference point on the shape curve denoted by s_{ref}.

attached to the head tip, frame $i \in \{0, \dots, N-1\}$ attached to each of the N joints, and finally the N frame attached to the tail tip. The output from the alignment algorithm is the origin and orientation of each of these frames. This output includes the angle q_i of each joint $i \in \{1, \dots, N\}$ of the VSR, which the motion planning framework outputs as reference angles for the physical robot.

3) The Placement of the Head Frame: The algorithm starts by defining the origin of the head frame as $\boldsymbol{O}_h = \boldsymbol{S}(s_h)$. Furthermore, the $\boldsymbol{x}_h$ axis is defined as illustrated in Fig. 3(a), i.e. as the unit vector from $\boldsymbol{O}_0$ to $\boldsymbol{O}_h$, where $\boldsymbol{O}_0$ is the origin of the frame of joint 1. The calculation of $\boldsymbol{O}_0$ is described below, so for now we assume that $\boldsymbol{O}_0$ has been defined. The $\boldsymbol{x}_h$ axis is therefore given as

$$\boldsymbol{x}_h = \frac{\boldsymbol{O}_h - \boldsymbol{O}_0}{\|\boldsymbol{O}_h - \boldsymbol{O}_0\|} \tag{5}$$

where we divide by the norm of the vector to make $\boldsymbol{x}_h$ a unit vector. As shown in Fig. 3(a), the $\boldsymbol{y}_h$ and $\boldsymbol{z}_h$ axes are determined by the roll angle ϕ_h of the VSR about the shape curve, which is an input to the algorithm. To this end, we first calculate a temporary axis denoted by $\boldsymbol{y}_h^{'}$, which we define to be horizontal through the cross product

$$\boldsymbol{y}_h^{'} = \frac{\boldsymbol{z}_g \times \boldsymbol{x}_h}{\|\boldsymbol{z}_g \times \boldsymbol{x}_h\|} \tag{6}$$

where $\boldsymbol{z}_g$ is the global z axis. A temporary $\boldsymbol{z}_h$ axis, denoted by $\boldsymbol{z}_h^{'}$, is then calculated as

$$\boldsymbol{z}_h^{'} = \boldsymbol{x}_h \times \boldsymbol{y}_h^{'} \tag{7}$$

The temporary head frame, which is defined by the axes $\boldsymbol{x}_h$, $\boldsymbol{y}_h^{'}$ and $\boldsymbol{z}_h^{'}$ as shown in Fig. 3(a), is now rotated by the specified angle ϕ_h about the $\boldsymbol{x}_h$ axis, thereby achieving the specified roll angle of the VSR about the shape curve. The orientation of the head frame with respect to the global frame is thus calculated as

$$\boldsymbol{R}_h^g = \begin{bmatrix}\boldsymbol{x}_h & \boldsymbol{y}_h & \boldsymbol{z}_h\end{bmatrix} = \begin{bmatrix}\boldsymbol{x}_h & \boldsymbol{y}_h^{'} & \boldsymbol{z}_h^{'}\end{bmatrix} \boldsymbol{R}_x(\phi_h) \tag{8}$$

where $\boldsymbol{R}_x$ is the rotation matrix for an elementary rotation about the x axis [3], which is defined as

$$\boldsymbol{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_h & -\sin\phi_h \\ 0 & \sin\phi_h & \cos\phi_h \end{bmatrix} \tag{9}$$

4) The Placement of the Joint Frames: The origin and orientation of the remaining $N+1$ coordinate frames are calculated in consecutive order by following the steps below for each frame. The origin of frame $i \in \{0, \dots, N\}$ is denoted by $\boldsymbol{O}_i$ and its coordinate axes are denoted by $\boldsymbol{x}_i$, $\boldsymbol{y}_i$ and $\boldsymbol{z}_i$, respectively. The algorithm makes use of a reference point s_{ref}, which is initialized as $s_{\text{ref}} = s_h$ and then traced backwards along the shape curve to define the *desired* placement of each frame.

To calculate $\boldsymbol{O}_i$, s_{ref} is moved backwards along the shape curve to the point whose linear distance to the previously aligned frame (i.e. $\boldsymbol{O}_{i-1}$) equals $l_{\text{LAD}} \in \mathbb{R}$. The design parameter l_{LAD} is referred to as the *look-ahead distance* and defines how far backwards along the shape curve to 'aim' the next link to be aligned. As shown in Fig. 3(b), the location of s_{ref} with respect to frame $i-1$ is defined as

$$\boldsymbol{r}_{\text{ref}} = \boldsymbol{S}(s_{\text{ref}}) - \boldsymbol{O}_{i-1} \tag{10}$$

where $\|\boldsymbol{r}_{\text{ref}}\| = l_{\text{LAD}}$. The algorithm attempts to place $\boldsymbol{O}_i$ such that the link between $\boldsymbol{O}_{i-1}$ and $\boldsymbol{O}_i$ (i.e. between joint i and joint $i+1$) is aligned with $\boldsymbol{r}_{\text{ref}}$. This link is, however, constrained to move in the plane normal to the rotation axis of joint i (i.e. $\boldsymbol{z}_{i-1}$), which means that a perfect alignment between the link and $\boldsymbol{r}_{\text{ref}}$ will generally not be possible. The algorithm therefore aligns the link such that the closest possible match with $\boldsymbol{r}_{\text{ref}}$ is achieved. As illustrated in Fig. 3(b), the closest match is found by first finding a vector $\boldsymbol{r}^{\perp}$ which is normal to the plane spanned by $\boldsymbol{z}_{i-1}$ and $\boldsymbol{r}_{\text{ref}}$, i.e.

$$\boldsymbol{r}^{\perp} = \boldsymbol{z}_{i-1} \times \boldsymbol{r}_{\text{ref}} \tag{11}$$

Subsequently, we find the closest match with $\boldsymbol{r}_{\text{ref}}$ by defining the link between $\boldsymbol{O}_{i-1}$ and $\boldsymbol{O}_i$ to be normal to the plane spanned by $\boldsymbol{r}^{\perp}$ and $\boldsymbol{z}_{i-1}$. As seen from Fig. 3(b), this is equivalent to defining the $\boldsymbol{x}_i$ axis as

$$\boldsymbol{x}_i = \boldsymbol{r}^{\perp} \times \boldsymbol{z}_{i-1} \tag{12}$$

We are now ready to calculate the angle q_i of joint i, which is the angle between the $\boldsymbol{x}_{i-1}$ and $\boldsymbol{x}_i$ axis (see Section II). We first express the $\boldsymbol{x}_i$ axis with respect to frame $i-1$. This vector is denoted by $\boldsymbol{x}_i^{i-1}$ and is found as

$$\boldsymbol{x}_i^{i-1} = \left(\boldsymbol{R}_{i-1}^g\right)^T \boldsymbol{x}_i \tag{13}$$

where $\boldsymbol{R}_{i-1}^g$ is the rotation matrix describing the global frame orientation of frame $i-1$, which was determined in the previous step of the algorithm. By denoting the x and

y component of this vector by $\boldsymbol{x}_{i,x}^{i-1}$ and $\boldsymbol{x}_{i,y}^{i-1}$, respectively, the joint angle is found as

$$q_i = \text{atan2}\left(\boldsymbol{x}_{i,y}^{i-1}, \boldsymbol{x}_{i,x}^{i-1}\right) \tag{14}$$

where atan2$(\cdot)$ is the four quadrant inverse tangent. Note that $\boldsymbol{x}_{i,z}^{i-1}$ is always zero since the $\boldsymbol{x}_i$ axis by design is normal to the $\boldsymbol{z}_{i-1}$ axis.

Using (2), we can now calculate the origin and orientation of frame i with respect to the global frame as

$$\boldsymbol{T}_i^g = \begin{bmatrix} \boldsymbol{x}_i & \boldsymbol{y}_i & \boldsymbol{z}_i & \boldsymbol{O}_i \\ 0 & 0 & 0 & 1 \end{bmatrix} = \boldsymbol{T}_{i-1}^g \boldsymbol{T}_i^{i-1}(q_i) \tag{15}$$

where $\boldsymbol{T}_{i-1}^g$ was determined in the previous step of the algorithm and $\boldsymbol{T}_i^{i-1}(q_i)$ is defined in (4).

Remark 3: The choice of look-ahead distance l_{LAD} is important. If we assume that all links of the robot have equal length, i.e. $a_i = a$ for $i \in \{0, \dots, N\}$, then l_{LAD} should be chosen such that $l_{\text{LAD}} \geq a$. If $l_{\text{LAD}} < a$, then the alignment will cause each link to 'overshoot' its reference point on the shape curve. From the authors' experience, the algorithm will generally perform well if $l_{\text{LAD}} \in [a, 2a]$.

Example 4: Fig. 4(b) shows the alignment of a VSR with $N = 6$ joints along the shape curve plotted in Fig. 4(a). The link lengths were $a_i = 0.1$ m, where $i \in \{0, \dots, 6\}$, and the look-ahead distance was $l_{\text{LAD}} = 2a_0$.

E. Generating Motion Patterns

1) General Approach: The presentation so far provides a framework for generating joint reference angles corresponding to body shapes defined by SCP coordinates. We can use this framework to generate dynamic motion patterns for a snake robot by varying the SCP coordinates and/or the location of the VSR along the shape curve with time. In particular, motion patterns can be generated in three ways:

1) Progressing the VSR forward along the shape curve while continuously retrieving its joint angles. The retrieved angles will vary as the VSR is progressed along the curve, thereby generating dynamic joint reference angles for the physical robot in accordance with the motion pattern drawn out by the shape curve.
2) Fixing the VSR on the shape curve while varying the SCP coordinates with time. Although the location of the VSR is fixed on the shape curve, its joint angles will vary as the SCP coordinates are changed.
3) A combination of approaches 1 and 2.

With approach 1, the VSR will at some point reach the last SCP of the shape curve. When this happens, we can maintain the motion by extending the curve with new SCPs according to the desired motion pattern. To this end, we introduce two important instruments. The first instrument is the *shape frame*, which defines the direction in which the shape curve is extended when new SCPs are added. The second instrument is the *gait segment*, which defines the shape curve over a single cycle of a motion pattern, thereby acting as the 'building block' of the shape curve. Section IV presents simulation results which illustrate these approaches.

2) The Shape Frame: When new SCPs are added in order to extend the shape curve, the direction in which the curve is extended will affect the direction in which the snake robot is propelled. We can utilize this property to steer the direction of the robot's motion by introducing a coordinate frame whose orientation is defined by a heading controller, and which is used as the reference frame when new SCPs are added to the shape curve. The frame is denoted as the *shape frame* and, as illustrated in Fig. 5, its axes are denoted by $\boldsymbol{x}_s$, $\boldsymbol{y}_s$ and $\boldsymbol{z}_s$, respectively. Moreover, its origin coincides with the last SCP of the shape curve, whose coordinates are $\boldsymbol{P}_{n-1} = \boldsymbol{S}(n-1)$. The orientation of the shape frame with respect to the global frame is expressed by the rotation matrix

$$\boldsymbol{R}_s^g = \begin{bmatrix} \boldsymbol{x}_s & \boldsymbol{y}_s & \boldsymbol{z}_s \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{16}$$

and can be regarded as a directional control input for the robot.

3) The Gait Segment: Snake robots are usually controlled according to predefined gait patterns, which are carried out in combination with feedback control laws that e.g. steer the heading and/or adapt the motion to the environment. To facilitate predefined gait patterns within the motion planning framework, we introduce a *gait segment*, which describes the shape curve over *one* cycle of a particular motion pattern. The gait segment is the 'building block' of the shape curve and is defined as a collection of k SCPs denoted by

$$\left\{\boldsymbol{P}_0^{\text{GS}}, \boldsymbol{P}_1^{\text{GS}}, \dots, \boldsymbol{P}_{k-1}^{\text{GS}}\right\} \tag{17}$$

where $\boldsymbol{P}_i^{\text{GS}} \in \mathbb{R}^3$ and superscript 'GS' is short for *gait segment*. Sustained motion according to the gait segment is achieved by repeatedly concatenating the shape curve with the gait segment coordinates while the VSR is progressed forward along the curve. The gait segment coordinates are expressed with respect to the shape frame. As illustrated in Fig. 5, an SCP from the gait segment is added to the shape curve by calculating its global frame coordinates $\boldsymbol{P}_{\text{new}}$ as

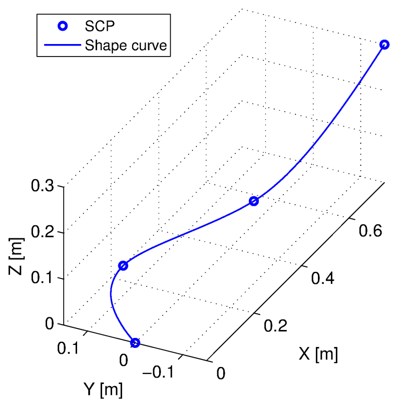
$$\boldsymbol{P}_{\text{new}} = \boldsymbol{P}_{n-1} + \boldsymbol{R}_s^g \left(\boldsymbol{P}_j^{\text{GS}} - \boldsymbol{P}_{j-1}^{\text{GS}}\right) \tag{18}$$

where $\boldsymbol{P}_{n-1}$ is the location of the last SCP of the shape curve, $\boldsymbol{R}_s^g$ is the rotation matrix that describes the orientation of the shape frame, and where $j \in \{1, \dots, k-1\}$. Note that the gait segment is concatenated with the shape curve *one* SCP at a time since this allows us to steer the direction of the motion by adjusting $\boldsymbol{R}_s^g$ each time a new SCP is added. After all k SCPs from the gait segment have been added to the shape curve, the process starts over from the first SCP.

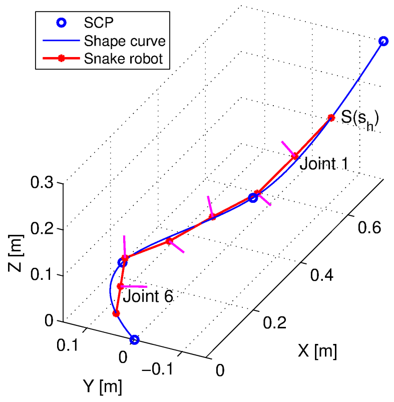
Example 5: A gait segment constructed from $k = 9$ SCPs is plotted in Fig. 4(c) using the same interpolation as in Example 1. The figure also shows the shape curve formed by concatenating the gait segment four times. The shape frame was initially oriented along the axes of the global frame. During the last two concatenations, however, the shape frame was rotated $45°$ about the global z axis by choosing $\boldsymbol{R}_s^g = \boldsymbol{R}_z(45°)$, where $\boldsymbol{R}_z$ describes an elementary rotation about the z axis. As a result, the progression direction of the shape curve was changed by $45°$.

IV. Applications of the Motion Planning Framework

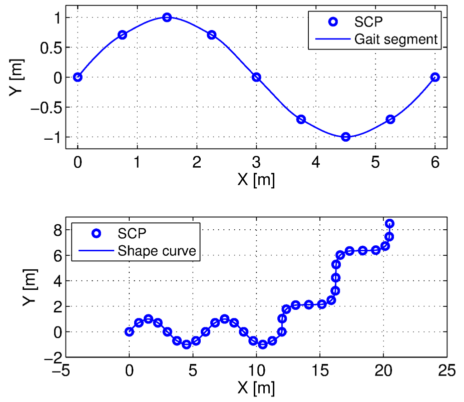
This section presents simulation results which demonstrate the applicability of the motion planning framework for realizing different types of three-dimensional motion patterns. The main feature to note from the simulation results is that at no point during the gait design process did we need to consider the specific joint angles of the robot.



(a) A shape curve constructed from $n = 4$ SCPs.



(b) A VSR with $N = 6$ joints aligned using $l_{\text{LAD}} = 2a_0$.



(c) A gait segment (top) and the resulting shape curve (bottom).

Fig. 4. (a) Example of a shape curve constructed from $n = 4$ SCPs. (b) Alignment of a VSR with $N = 6$ joints using the look-ahead distance $l_{\text{LAD}} = 2a_0$. The transversal axis at each joint indicates the rotation axis. (c) An example of a gait segment (top) constructed from $k = 9$ SCPs and the shape curve (bottom) formed by concatenating the gait segment four times. During the last two concatenations, the shape frame was rotated 45° about the z axis.

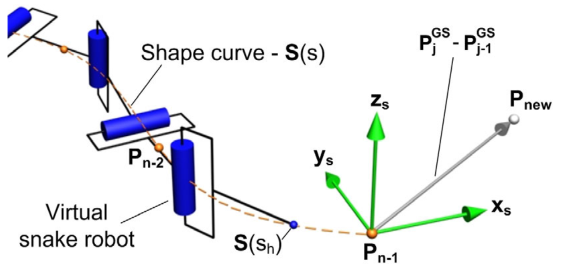


Fig. 5. New SCPs are added to the shape curve with reference to the *shape frame*, whose origin coincides with the last SCP of the curve.

A. Simulation Setup

The simulations were carried out using *Open Dynamics Engine* (ODE) [10], which is an open-source software library for simulating articulated rigid-body dynamics. The software allows for fast simulations of complex articulated structures in easily reconfigurable obstacle environments.

The snake robot was implemented in ODE according to the kinematics in Fig. 1. The number of joints was $N = 16$ and the distance between the joints was $a_i = 0.08$ m for $i \in \{0, \ldots, N\}$. The robot moved on a flat surface and had a quadratic cross-section with height/width equal to 0.07 m.

The motion planning framework was implemented in *Matlab R2011a* and the kinematic parameters of the VSR were equal to those of the robot in ODE. Shape curves were constructed using the piecewise cubic hermite interpolating polynomial described in Example 1. Moreover, the alignment of the VSR to the shape curve was carried out with look-ahead distance $l_{\text{LAD}} = 2a_0 = 0.16$ m. The joint angles of the VSR were used as joint reference angles for the snake robot in ODE with an update frequency of 30 Hz.

B. Simulation 1: Sidewinding Motion

During sidewinding motion, a snake moves sideways by raising its head and throwing it sideways, and then repeating the same motion with the rest of its body in a cyclic manner [2]. A snake robot can achieve this motion by moving its body according to a horizontal wave superimposed by a vertical wave with a 90° phase shift between the two waves. To this end, we first specified the desired body shape of the snake robot by defining the parametric curve

$$\boldsymbol{B}(\beta) = \begin{bmatrix} B_x(\beta) \\ B_y(\beta) \\ B_z(\beta) \end{bmatrix} = \begin{bmatrix} k_x \frac{\beta}{2\pi} \\ k_y \sin(\beta) \\ k_z \sin(\beta + \frac{\pi}{2}) \end{bmatrix} \in \mathbb{R}^3 \tag{19}$$

where $\beta \in [0, 2\pi]$, $k_x \in \mathbb{R}$ defines the length of the curve along the x axis, $k_y \in \mathbb{R}$ defines the amplitude of the horizontal wave in the x-y plane, and $k_z \in \mathbb{R}$ defines the amplitude of the vertical wave (phase shifted by 90°) in the x-z plane. We then used $\boldsymbol{B}(\beta)$ to define a gait segment with $k = 9$ SCPs according to (17) such that

$$\boldsymbol{P}_i^{\text{GS}} = \boldsymbol{B}(\tfrac{2\pi}{8} i) \quad , \quad i \in \{0, \ldots, 8\} \tag{20}$$

The resulting gait segment and the aligned VSR are plotted in Fig. 6(a), where we chose $k_x = 0.7a_0(N+1) = 0.952$ m, $k_y = 3a_0 = 0.24$ m, and $k_z = a_0/3 = 0.0267$ m.

To achieve sidewinding motion, we followed the approach described in Section III-E.3 and progressed the VSR forward along the shape curve at a constant speed such that $\left|\dot{\boldsymbol{S}}(s_h)\right| = 0.5$ m/s and with roll angle $\phi_h = 0$. Furthermore, to illustrate how the shape frame described in Section III-E.2 can be used to influence the direction of the motion, the shape frame was first oriented along the axes of the global frame and then rotated at a constant velocity about the global z axis in order to steer the robot in the counter-clockwise direction. In particular, the shape frame in (16) was defined as $\boldsymbol{R}_s^g = \boldsymbol{R}_z(\psi_s)$, where $\boldsymbol{R}_z(\psi_s)$ is the rotation matrix for an elementary rotation about the global z axis by an angle ψ_s, which we defined as $\psi_s = 0°$ and $\dot{\psi}_s = 0°/s$ for $t \in [0\text{ s}, 5\text{ s}]$, $\dot{\psi}_s = 22.5°/s$ for $t \in [5\text{ s}, 10\text{ s}]$, and $\dot{\psi}_s = 0°/s$ for $t \in [10\text{ s}, 15\text{ s}]$.

The shape curve developed during 15 seconds of motion is plotted in Fig. 6(b), while Fig. 6(c) shows the position of the centre link of the snake robot (link 9) simulated using ODE.

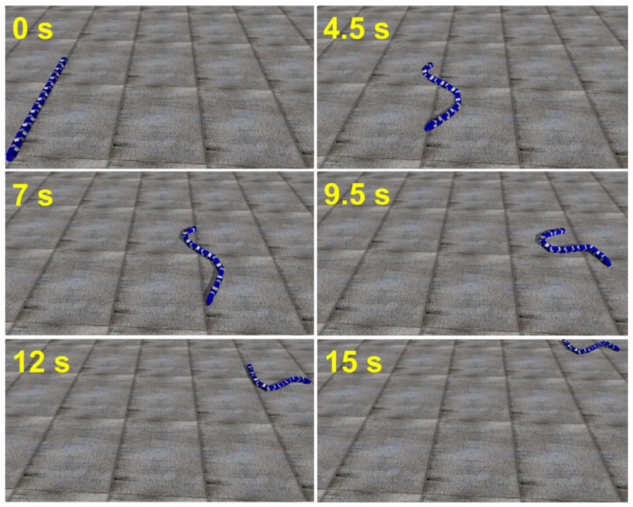
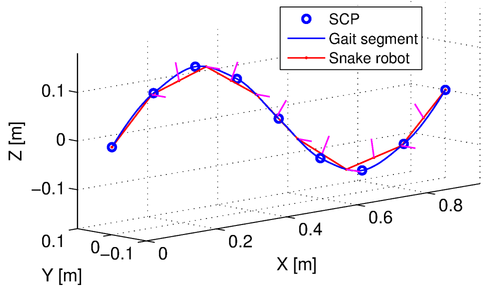


Fig. 7. Screen-shots of sidewinding motion simulated in ODE.

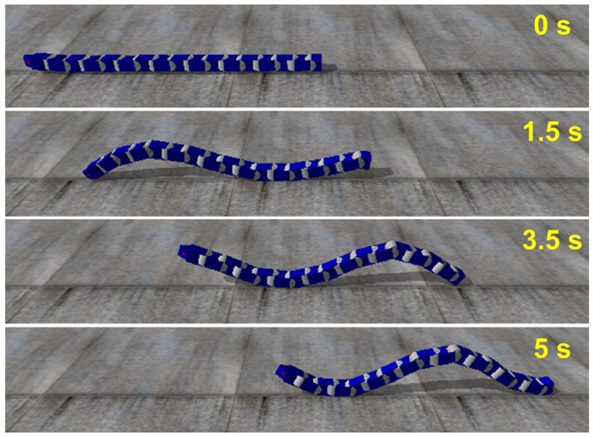
The ODE simulation is also visualized in Fig. 7. The results show that sidewinding motion was successfully achieved and that the shape frame rotation successfully steered the robot in the counter-clockwise direction.

C. Simulation 2: Vertical Wave Motion

The next simulation illustrates how propulsion can be achieved by vertical body waves. To this end, we followed the exact same approach as in the previous subsection with the following differences. The parametric curve in (19) was redefined to be a purely vertical wave in the x-z plane by defining $B_x(\beta) = k_x \frac{\beta}{2\pi}$, $B_y = 0$ and $B_z(\beta) = k_z \sin(\beta)$, where we chose $k_x = 0.7a_0(N+1) = 0.952$ m and $k_z = 0.1a_0(N+1) = 0.136$ m. We also increased the speed of the VSR along the shape curve such that $\left|\dot{\boldsymbol{S}}(s_h)\right| = 2$ m/s, and we defined the shape frame to be aligned with the global frame ($\boldsymbol{R}_s^g = \boldsymbol{I}_{3\times3}$). The simulation result is shown in Fig. 8, where Fig. 8(a) shows the resulting gait segment and the aligned VSR, while Fig. 8(b) shows screen-shots from the ODE simulation which illustrate how the vertical body waves propelled the robot.



(a) The gait segment.

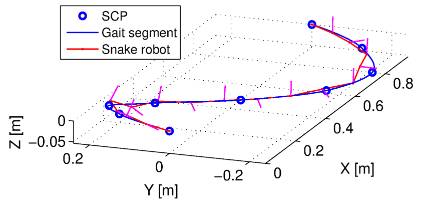


(b) Screen-shots from ODE.

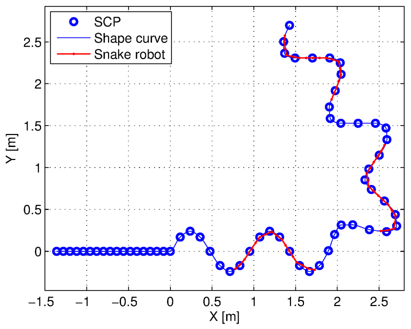
Fig. 8. Simulation results of vertical wave motion. (a) The gait segment and the aligned VSR, where the transversal axis at each joint indicates the rotation axis. (b) Screen-shots of the motion simulated in ODE.

D. Simulation 3: Lateral Rolling Motion

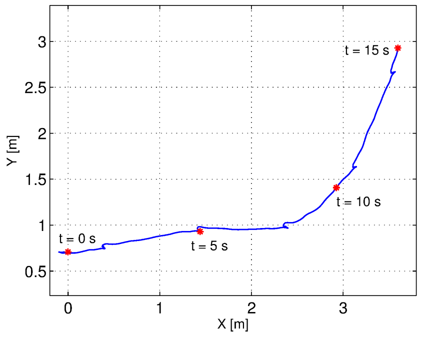
During lateral rolling motion, a snake robot moves sideways by curving its body slightly while rolling about its longitudinal axis. Instead of progressing the VSR forward along the shape curve by continuously increasing s_h, we can achieve this rolling motion by keeping s_h fixed while continuously increasing the roll angle ϕ_h about the curve. In particular, we defined a constant shape curve consisting



(a) The gait segment.

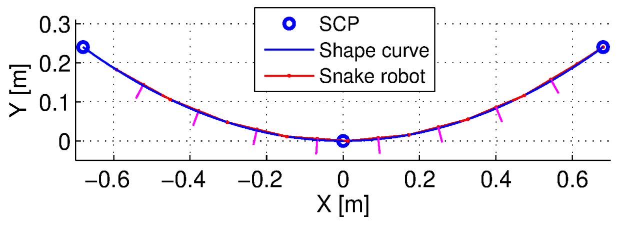


(b) The shape curve.

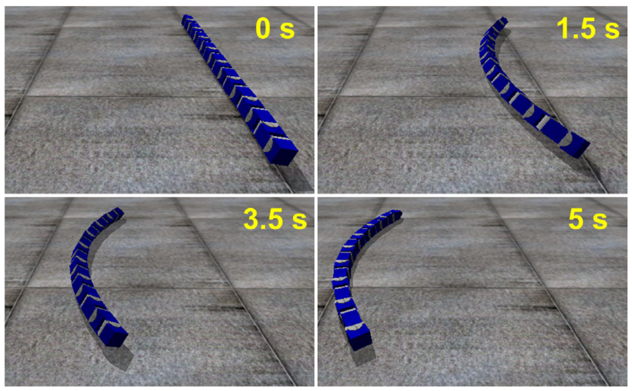


(c) The position of the snake robot.

Fig. 6. Simulation results of sidewinding motion. (a) The gait segment and the aligned VSR, where the transversal axis at each joint indicates the rotation axis. (b) The shape curve developed during the motion, where the aligned VSR is plotted at $t = 5$ s, 10 s, and 15 s, respectively. (c) The position of the centre link (link 9) of the snake robot simulated using ODE.



(a) The shape curve.



(b) Screen-shots from ODE.

Fig. 9. Simulation results of lateral rolling motion. (a) The shape curve and the aligned VSR, where the transversal axis at each joint indicates the rotation axis. (b) Screen-shots of the motion simulated in ODE.

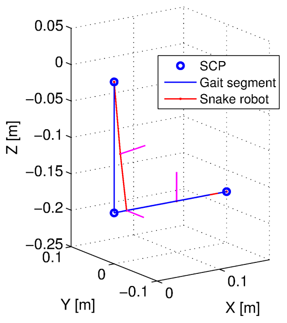
of the three SCPs $\boldsymbol{P}_0 = (-0.5a_0(N+1), 3a_0, 0) = (-0.68, 0.24, 0)$, $\boldsymbol{P}_1 = (0,0,0)$, $\boldsymbol{P}_2 = (0.5a_0(N+1), 3a_0, 0) = (0.68, 0.24, 0)$. Furthermore, we fixed the head of the VSR at the last SCP by defining $s_h = 2$ and we continuously rotated the VSR about the shape curve by defining $\dot{\phi}_h = -360°/s$. The simulation result is shown in Fig. 9, where Fig. 9(a) shows the shape curve and the aligned VSR, while Fig. 9(b) shows screen-shots from the ODE simulation which illustrate the sideways rolling motion.

E. Simulation 4: Descending a Staircase

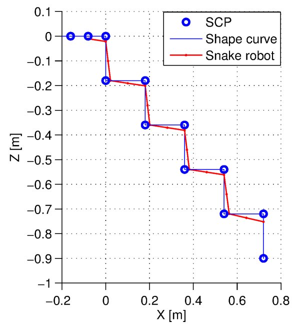
The last simulation illustrates the explicit nature of the motion planning framework in terms of defining the body shape of the snake robot. The goal was to make the snake robot in ODE crawl down a staircase with steps that were 0.2 m in both depth and height. This was achieved by defining a gait segment corresponding to a single step of the staircase as shown in Fig. 10(a). As a result, the progression of the VSR along the shape curve, which is plotted in Fig. 10(b) at $t = 7.5$ s, caused a staircase-like pattern to be propagated along the body of the simulated robot. This pattern enabled the robot to climb stepwise down the staircase as illustrated in Fig. 10(c).

V. Conclusions

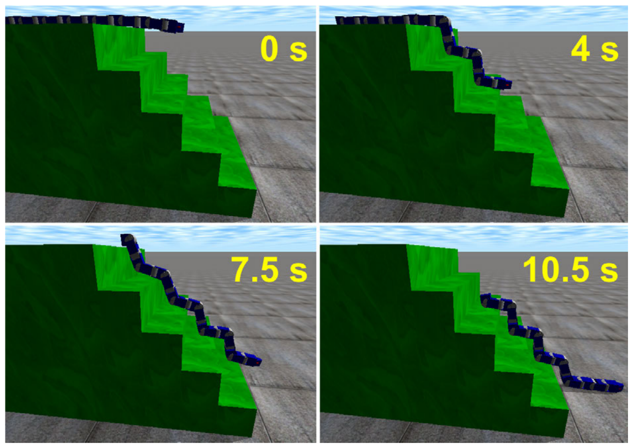
This paper has presented a motion planning framework for three-dimensional body shape control of snake robots, where continuous curves defined by shape control points are mapped to dynamic motion patterns for the snake robot. The framework allows the motion to be controlled by parameters which are intuitively mapped to the overall body shape of the snake robot. The applicability of the framework was demonstrated by simulation results.



(a) The gait segment.



(b) The shape curve.



(c) Screen-shots from ODE.

Fig. 10. Simulation results of a snake robot descending a staircase. (a) The gait segment and a few joints of the aligned VSR, where the transversal axis at each joint indicates the rotation axis. (b) The shape curve developed during the motion, where the aligned VSR is plotted at $t = 7.5$ s. (c) Screen-shots of the motion simulated in ODE.

References

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

[2] P. Liljebäck, K. Y. Pettersen, Ø. Stavdahl, and J. T. Gravdahl, *Snake Robots - Modelling, Mechatronics, and Control*, ser. Advances in Industrial Control. Springer, 2013.

[3] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Springer, 2008.

[4] G. S. Chirikjian, "Design and analysis of some nonanthropomorphic, biologically inspired robots: An overview," *Journal of Robotic Systems*, vol. 18, no. 12, pp. 701–713, December 2001.

[5] 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.

[6] 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.

[7] 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.

[8] P. Liljebäck, K. Y. Pettersen, Ø. Stavdahl, and J. T. Gravdahl, "Compliant control of the body shape of snake robots," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2014, pp. 4548–4555.

[9] F. N. Fritsch and R. E. Carlson, "Monotone piecewise cubic interpolation," *SIAM J. Numerical Analysis*, vol. 17, no. 2, pp. 238–246, 1980.

[10] R. Smith, "Open Dynamics Engine," http://www.ode.org, 2014, online.