J Intell Robot Syst (2014) 75:17–28
DOI 10.1007/s10846-013-9858-y

Dynamics and Motion Planning of Trident Snake Robot

Zuzanna Pietrowska · Krzysztof Tchoń

Received: 17 January 2013 / Accepted: 4 July 2013 / Published online: 20 July 2013

Abstract The trident snake robot is a mechanical device that serves as a demanding testbed for motion planning and control algorithms of constrained non-holonomic systems. This paper provides the equations of motion and addresses the motion planning problem of the trident snake with dynamics, equipped with either active joints (undulatory locomotion) or active wheels (wheeled locomotion). Thanks to a partial feedback linearization of the dynamics model, the motion planning problem basically reduces to a constrained kinematic motion planning. Two kinds of constraints have been taken into account, ensuring the regularity of the feedback and the collision avoidance between the robot's arms and body. Following the guidelines of the endogenous configuration space approach, two Jacobian motion planning algorithms have been designed: the singularity robust Jacobian algorithm and the imbalanced Jacobian algorithm. Performance of these algorithms have been illustrated by computer simulations.

Keywords Non-holonomic system · Dynamics · Motion planning · Trident snake

1 Introduction

The snake-like robots belong to biologically inspired robotic devices of remarkable potential of applicability. These robots have been studied for more than two decades, with focus on their locomotion and manipulation capabilities [1]. The state of the art in kinematic and dynamics modeling of snake-like robots has been presented exhaustively in [2]. The trident snake is a specific snake-like robot invented out of purely theoretical interest in non-holonomic locomotion, and used as a testbed for motion planning and control algorithms of non-holonomic systems [3]. A challenge in the control of trident snake results from the existence of complex singular configurations and possible collisions of robot's arms with its body. The complexity of to be avoided configurations explodes along with making the robot's arms multiple-links. Existing motion planning algorithms for trident snake primarily restrict to the kinematics of single or double-link arms. These algorithms have been derived using either geometric

The work of Krzysztof Tchoń was supported by the Wrocław University of Technology under a statutory grant.

Z. Pietrowska · K. Tchoń (✉)
Institute of Computer Engineering, Control and
Robotics, Wrocław University of Technology,
ul. Janiszewskiego 11/17, 50–372 Wrocław, Poland
e-mail: krzysztof.tchon@pwr.wroc.pl

Z. Pietrowska
e-mail: 163094@student.pwr.wroc.pl

methods [4–6] or the endogenous configuration space approach [7–9]. Trajectory tracking algorithms for trident snake, based on the transverse function method, have been proposed in [10]. In [11] the transverse function method is applied to a non-symmetric trident snake, with two single-link and one double-link arm. The trident snake robot designed in our robotics laboratory is shown in Fig. 1, and described in [12].

Differently to the mentioned works that concentrate on the kinematics of the trident snake with undulatory locomotion, this paper addresses the motion planning problem of the trident snake robot with dynamics, equipped with either actuated joints (undulatory locomotion) or actuated wheels (wheeled locomotion). In the former case, the robot's motion is subordinated to nonholonomic constraints preventing only the sliding of the wheels, in the latter, both the sliding and the slipping of the wheels are not permitted. By its very nature, the motion planning problem of the trident snake includes additional constraints guaranteeing the avoidance of singular configurations and collisions between the robot's arms and its body. In what follows this problem will be solved by applying the endogenous configuration space approach [13, 14] to partially feedback linearized equations of motion. Two Jacobian motion planning algorithms, the singularity robust Jacobian algorithm, Tchoń and Jakubiak [13] and the imbalanced Jacobian algorithm, Janiak and Tchoń [15], are utilized. Performance of these algorithms is illustrated by computer simulations. A dynamics model of another trident snake-like device, called the trident steering walker, has been recently studied in [16].

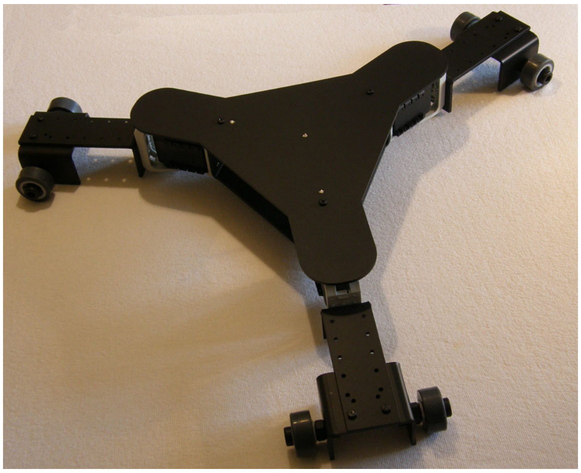


Fig. 1 Trident snake robot

The organization of this paper is the following. Section 2 presents the Lagrangian equations of motion of the trident snake. The motion planning problem is stated and solved in Section 3. Section 4 contains results of numeric computations. The paper is concluded with Section 5.

2 Equations of Motion

A geometric picture of the trident snake with attached coordinate frames is displayed in Fig. 2. The robot's body resembles a triangle inscribed in a circle of radius r. The robot has three arms of length l, fixed to the body at its vertexes by revolute joints and supported by wheels of radius R_k. The robot moves on a horizontal plane, in response to either joints or wheels actuation. In the former case the wheels are assumed to not slide, in the latter case both the sliding as well as the slipping of the wheels is not permitted. The meaning of variables and geometric parameters describing the trident snake have been defined in the figure, note that $\alpha_2 = 0$.

2.1 Constrained System Dynamics

Suppose that the motion of a robotic system is characterized by generalized coordinates and velocities $(q, \dot{q}) \in R^n \times R^n$, and subject to $l < n$ independent phase constraints in the form of Pfaff

$$A(q)\dot{q} = 0, \tag{1}$$

where $A(q)$ is an $l \times n$ matrix of rank l. Under these assumptions the system's kinematics can be represented as a driftless control system

$$\dot{q} = G(q)\eta = \sum_{i=1}^{m} g_i(q)\eta_i, \tag{2}$$

with control vector $\eta = (\eta_1, \eta_2, \ldots, \eta_m)^T \in R^m$, $m = n - l$. The control matrix of system (2),

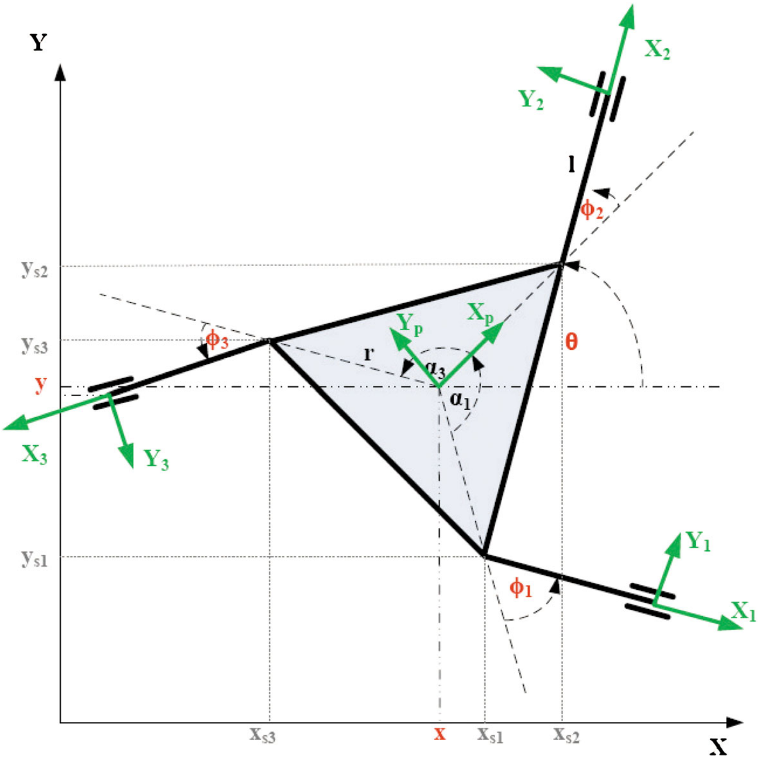


Fig. 2 Trident snake robot: a geometric scheme

$G(q) = \left[g_1(q), g_2(q), \ldots, g_m(q)\right]$, satisfies the condition $A(q)G(q) = 0$. The model of system's dynamics can be derived from the d'Alembert principle. Let $L(q, \dot{q})$ denote the Lagrangian of the free (unconstrained) system. Then, the Euler–Lagrange equations assume the form

$$\begin{aligned} \frac{d}{dt}\frac{\partial L(q,\dot{q})}{\partial \dot{q}} - \frac{\partial L(q,\dot{q})}{\partial q} &= Q(q)\ddot{q} + C(q,\dot{q})\dot{q} + D(q) \\ &= F(q,\dot{q}) + B(q)u. \end{aligned} \tag{3}$$

The right hand side of Eq. 3 contains traction forces that by the d'Alembert are equal to $F(q, \dot{q}) = A^T(q)\lambda$, with Lagrange multipliers $\lambda \in R^l$, and control forces (torques) $u \in R^r, r \leq l$, pre-multiplied by a control matrix $B(q)$. A standard elimination of the multipliers results in the following equations of motion

$$\dot{q} = G(q)\eta, \;\; \dot{\eta} = N(q, \eta) + P(q)u, \quad y = h(q, \eta), \tag{4}$$

where $y \in R^p$ denotes output variables subject to the motion planing. The matrices and vectors appearing in Eq. 4 are defined as follows

$$\begin{aligned} M(q) &= G^T(q)Q(q)G(q), \\ P(q) &= M^{-1}(q)G^T(q)B(q), \\ N(q, \eta) &= -M^{-1}(q)G^T(q)((Q(q)\dot{G}(q) \\ &\quad + C(q, G(q)\eta)G(q))\eta + D(q)). \end{aligned}$$

In the next subsections the equations (4) will be specified to the dynamics of the trident snake robot with either actuated (active) joints and non-actuated (passive) wheels or non-actuated (passive) joints and actuated (active) wheels.

2.2 Trident Snake: Passive Wheels Without Sliding

2.2.1 Kinematics

The trident snake robot with active joints and passive wheels can be described by generalized coordinates $q = (x, y, \theta, \phi_1, \phi_2, \phi_3)^T \in R^6$. Coordinates (x, y, θ) refer to the robot's position and orientation, the remaining coordinates define positions of the joints. It is easily checked that the Pfaff matrix (1) corresponding to the motion of wheels without sliding takes the form

$$A(q) = \left[A_1(\phi) Rot^T(Z, \theta) \; -lI_3 \right], \tag{5}$$

where

$$A_1(\phi) = \begin{bmatrix} \sin(\alpha_1 + \phi_1) & -\cos(\alpha_1 + \phi_1) & -l - r\cos\phi_1 \\ \sin(\alpha_2 + \phi_2) & -\cos(\alpha_2 + \phi_2) & -l - r\cos\phi_2 \\ \sin(\alpha_3 + \phi_3) & -\cos(\alpha_3 + \phi_3) & -l - r\cos\phi_3 \end{bmatrix}, \tag{6}$$

and $Rot(Z, \theta)$ stands for the rotation matrix about the Z axis by the angle θ. Consequently, the robot's kinematics will be represented by the control system (2), with the control matrix

$$G(q) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \\ \frac{\sin(\alpha_1 + \phi_1)}{l} & -\frac{\cos(\alpha_1 + \phi_1)}{l} & -1 - \frac{r\cos\phi_1}{l} \\ \frac{\sin(\alpha_2 + \phi_2)}{l} & -\frac{\cos(\alpha_2 + \phi_2)}{l} & -1 - \frac{r\cos\phi_2}{l} \\ \frac{\sin(\alpha_3 + \phi_3)}{l} & -\frac{\cos(\alpha_3 + \phi_3)}{l} & -1 - \frac{r\cos\phi_3}{l} \end{bmatrix} = \begin{bmatrix} G_1(\theta) \\ G_2(\phi) \end{bmatrix}, \tag{7}$$

both the upper and the lower block being 3 × 3 matrices.

2.2.2 Dynamics

Assuming the notation: M_k—the mass of each wheel, M_0—the mass of the body, d—the thickness of each wheel, $I_0 = \frac{1}{4}M_0 r^2$, $I_{0k} = \frac{1}{4}M_k R_k^2 + \frac{1}{12}M_k d^2$—moments of inertia, we compute the body's kinetic energy as

$$E_{kk} = \frac{1}{2}M_0(\dot{x}^2 + \dot{y}^2) + \frac{1}{2}I_0\dot{\theta}^2, \tag{8}$$

and the ith wheel kinetic energy as

$$\begin{aligned} E_{ki} = {} & \frac{1}{2}I_{0k}(\dot{\phi}_i + \dot{\theta})^2 \\ & + \frac{1}{2}M_k\big(\dot{x}^2 + \dot{y}^2 + l^2(\dot{\phi}_i + \dot{\theta})^2 + r^2\dot{\theta}^2 \\ & + 2rl(\dot{\phi}_i + \dot{\theta})\dot{\theta}\cos\phi_i + 2l(\dot{\phi}_i + \dot{\theta}) \\ & \times \big(\dot{y}\cos(\alpha_i + \phi_i + \theta) - \dot{x}\sin(\alpha_i + \phi_i + \theta)\big) \\ & + 2r\dot{\theta}\big(\dot{y}\cos(\alpha_i + \theta) - \dot{x}\sin(\alpha_i + \theta)\big)\big). \end{aligned} \tag{9}$$

For the reason that the robot moves on a horizontal plane, its potential energy is set to zero, so the Lagrangian

$$L_1(q, \dot{q}) = E_{kk} + \sum_{i=1}^{3} E_{ki}. \tag{10}$$

Taking into account the kinematics model (7) and the form of the Lagrangian (10), we obtain the following equations of motion of the trident snake robot with active joints and passive wheels

$$\begin{aligned} \dot{q} &= G(q)\eta, \\ \dot{\eta} &= N(q, \eta) + P(q)u, \\ y &= h(q, \eta) = (q, \eta). \end{aligned} \tag{11}$$

Components of the vector $N(q, \eta)$ and entries of the matrix $P(q)$ have been computed symbolically using MATHEMATICA. The interested reader is referred to [17] for details. Due to the actuation of the robot's joints, the control matrix $B(q) = [0_3, I_3]^T$.

2.3 Trident Snake: Active Wheels Without Sliding and Slipping

2.3.1 Kinematics

The trident snake with passive joints and active wheels can be characterized by the generalized coordinates $q = (x, y, \theta, \phi_1, \phi_2, \phi_3, \beta_1, \beta_2, \beta_3)^T \in R^9$, where the last three components denote the revolution angles of the wheels. Under assumption that the wheels neither slide nor slip, the Pfaffian matrix (1) is

$$A(q) = \begin{bmatrix} A_1(\phi) Rot^T(Z, \theta) & -lI_3 & 0_3 \\ A_2(\phi, \theta) & 0_3 & -R_k I_3 \end{bmatrix}, \tag{12}$$

where the matrix $A_1(\phi)$ has already been defined by Eq. 6, and the matrix

$$A_2(\phi,\theta)=\begin{bmatrix}\cos(\alpha_1+\phi_1+\theta) & \sin(\alpha_1+\phi_1+\theta) & r\sin\phi_1\\ \cos(\alpha_2+\phi_2+\theta) & \sin(\alpha_2+\phi_2+\theta) & r\sin\phi_2\\ \cos(\alpha_3+\phi_3+\theta) & \sin(\alpha_3+\phi_3+\theta) & r\sin\phi_3\end{bmatrix}. \tag{13}$$

As a consequence, the control system (2) is determined by the control matrix

$$G(q)=\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\\ \frac{\sin(\alpha_1+\phi_1)}{l} & -\frac{\cos(\alpha_1+\phi_1)}{l} & -1-\frac{r\cos\phi_1}{l}\\ \frac{\sin(\alpha_2+\phi_2)}{l} & -\frac{\cos(\alpha_2+\phi_2)}{l} & -1-\frac{r\cos\phi_2}{l}\\ \frac{\sin(\alpha_3+\phi_3)}{l} & -\frac{\cos(\alpha_3+\phi_3)}{l} & -1-\frac{r\cos\phi_3}{l}\\ \frac{\cos(\alpha_1+\phi_1)}{R_k} & \frac{\sin(\alpha_1+\phi_1)}{R_k} & \frac{r\sin\phi_1}{R_k}\\ \frac{\cos(\alpha_2+\phi_2)}{R_k} & \frac{\sin(\alpha_2+\phi_2)}{R_k} & \frac{r\sin\phi_2}{R_k}\\ \frac{\cos(\alpha_3+\phi_3)}{R_k} & \frac{\sin(\alpha_3+\phi_3)}{R_k} & \frac{r\sin\phi_3}{R_k}\end{bmatrix}$$

$$=\begin{bmatrix}G_1(\theta)\\ G_2(\phi)\\ T(\phi)\end{bmatrix}, \tag{14}$$

partitioned into three 3 × 3 blocks, two upper of which coinciding with Eq. 7.

2.3.2 Dynamics

The expression (8) for the kinetic energy of the body remains unchanged. The kinetic energy of a wheel (9) should be complemented with a component $E_{kti}=\frac{1}{2}I_{kz}\dot{\beta}_i^2$ referring to its rolling, where $I_{kz}=\frac{1}{2}M_kR_k^2$ denotes the wheel's moment of inertia with respect to its rotation axis. Additionally, to improve the accuracy of dynamics modelling, the kinetic energies of the links and the actuators need to be taken into account. To this aim, we let M_l and M_s denote, respectively, the mass of a single link and of the actuator mounted at the end of each link. It will be assumed that the links are homogeneous, thin bars, and the actuators are point masses attached to the links. Then, treating the robot's arm as simple manipulators, and using a standard modeling procedure, we obtain the kinetic energy of the i-th arm equal to

$$\begin{aligned}E_{kl_i}=\frac{1}{6}M_l\Big(&l^2\dot{\phi}_i^2+(l^2+3r^2+3lr\cos\phi_i)\dot{\theta}^2\\ &+l\dot{\phi}_i\big((2l+3r\cos\phi_i)\dot{\theta}-3\sin(\alpha_i+\phi_i+\theta)\dot{x}\\ &+3\cos(\alpha_i+\phi_i+\theta)\dot{y}\big)+\dot{\theta}(-3(2r\sin(\alpha_i+\theta)\\ &+l\sin(\alpha_i+\phi_i+\theta))\dot{x}+3(2r\cos(\alpha_i+\theta)\\ &+l\cos(\alpha_i+\phi_i+\theta))\dot{y})+3(\dot{x}^2+\dot{y}^2)\Big).\end{aligned}$$

and get the expression for the total kinetic energy of the actuators

$$\begin{aligned}E_{ac}=\frac{1}{2}M_s\big(&l^2\dot{\phi}_1^2+l^2\dot{\phi}_2^2+l^2\dot{\phi}_3^2\\ &+2l^2\dot{\phi}_3\dot{\theta}+2lr\cos\phi_3\dot{\phi}_3\dot{\theta}+3l^2\dot{\theta}^2+3r^2\dot{\theta}^2\\ &+2lr\cos\phi_1\dot{\theta}^2+2lr\cos\phi_2\dot{\theta}^2+2lr\cos\phi_3\dot{\theta}^2\\ &-2l\sin(\alpha_3+\phi_3+\theta)\dot{\phi}_3\dot{x}-2r\sin(\alpha_1+\theta)\dot{\theta}\dot{x}\\ &-2r\sin(\alpha_2+\theta)\dot{\theta}\dot{x}-2r\sin(\alpha_3+\theta)\dot{\theta}\dot{x}\\ &-2l\sin(\alpha_1+\phi_1+\theta)\dot{\theta}\dot{x}\\ &-2l\sin(\alpha_2+\phi_2+\theta)\dot{\theta}\dot{x}\\ &-2l\sin(\alpha_3+\phi_3+\theta)\dot{\theta}\dot{x}\\ &+3\dot{x}^2+2l\cos(\alpha_3+\phi_3+\theta)\dot{\phi}_3\dot{y}\\ &+2r\cos(\alpha_1+\theta)\dot{\theta}\dot{y}+2r\cos(\alpha_2+\theta)\dot{\theta}\dot{y}\\ &+2r\cos(\alpha_3+\theta)\dot{\theta}\dot{y}+2l\cos(\alpha_1+\phi_1+\theta)\dot{\theta}\dot{y}\\ &+2l\cos(\alpha_2+\phi_2+\theta)\dot{\theta}\dot{y}\\ &+2l\cos(\alpha_3+\phi_3+\theta)\dot{\theta}\dot{y}+3\dot{y}^2\\ &+2l\dot{\phi}_1((l+r\cos\phi_1)\dot{\theta}-\sin(\alpha_1+\phi_1+\theta)\dot{x}\\ &+\cos(\alpha_1+\phi_1+\theta)\dot{y})+2l\dot{\phi}_2((l+r\cos\phi_2)\dot{\theta}\\ &-\sin(\alpha_2+\phi_2+\theta)\dot{x}+\cos(\alpha_2+\phi_2+\theta)\dot{y})\big).\end{aligned}$$

Eventually, the Lagrangian of the free system

$$L_2(q,\dot{q})=E_{kk}+\sum_{i=1}^{3}(E_{ki}+E_{kti}+E_{kl_i})+E_{ac}. \tag{15}$$

Following the guidelines presented above, we obtain the equations of motion of the trident snake with active wheels in the form

$$\begin{aligned}\dot{q} &= G(q)\eta,\\ \dot{\eta} &= N(q,\eta) + P(q)u,\\ y &= h(q,\eta) = (x, y, \theta, \phi, \eta).\end{aligned} \tag{16}$$

Our choice of the output variables means that the revolution angles of the wheels $(\beta_1, \beta_2, \beta_3)$ will not be subject to motion planning. As previously, components of the vector $N(q,\eta)$ and entries of the matrix $P(q)$ have been computed symbolically; detailed formulas again can be found in [17]. Observe that the actuation of wheels implies that the control matrix $B(q) = [0_3, 0_3, I_3]^T$.

3 Motion Planning

We shall study the following motion planning problem: Given the equations of motion (4) and an initial state (q_0, η_0), find a control $u(t) \in R^r$, able to drive the system's output to a desired point y_d at prescribed $T > 0$, respecting all the constraints imposed on the system.

The motion planning algorithm will be based on the endogenous configuration space approach [13, 14]. Its derivation will be accomplished for the general system (4), and then specified to models (11) and (16). To begin, let us observe that the matrix $P(q)$ in Eq. 4 is square, and invertible, if only the matrix $\Gamma(q) = G^T(q)B(q)$ is such. Singularities of this matrix for active joints and for active wheels are depicted in Fig. 3, see [8]. The plots show complexity of the singular set. Assuming invertibility of the matrix $\Gamma(q) = G^T(q)B(q)$, we can apply a feedback

$$u = -P^{-1}(q)N(q,\eta) + P^{-1}(q)v \tag{17}$$

transforming the system (4) to the form

$$\dot{q} = G(q)\eta, \;\; \dot{\eta} = v, \quad y = h(q,\eta), \tag{18}$$

that can be referred to as kinematic [18]. This kinematic reduction results in the following approach to the motion planning problem: First find a solution $(v(t), q(t), \eta(t))$ of the motion planning problem with constraints in the system (18), guaranteeing invertibility of $\Gamma(q)$ and preventing collisions between the robot's body and arms, and then substitute this solution into the feedback (17) to get the solution $u(t)$ of the original motion planning problem in the system (4). The advantage of such an approach consists in the fact that solving a constraint motion planning problem in the kinematic system (18) is much simpler than in the original system. The constraints that the trajectory of Eq. 18 should satisfy take the form

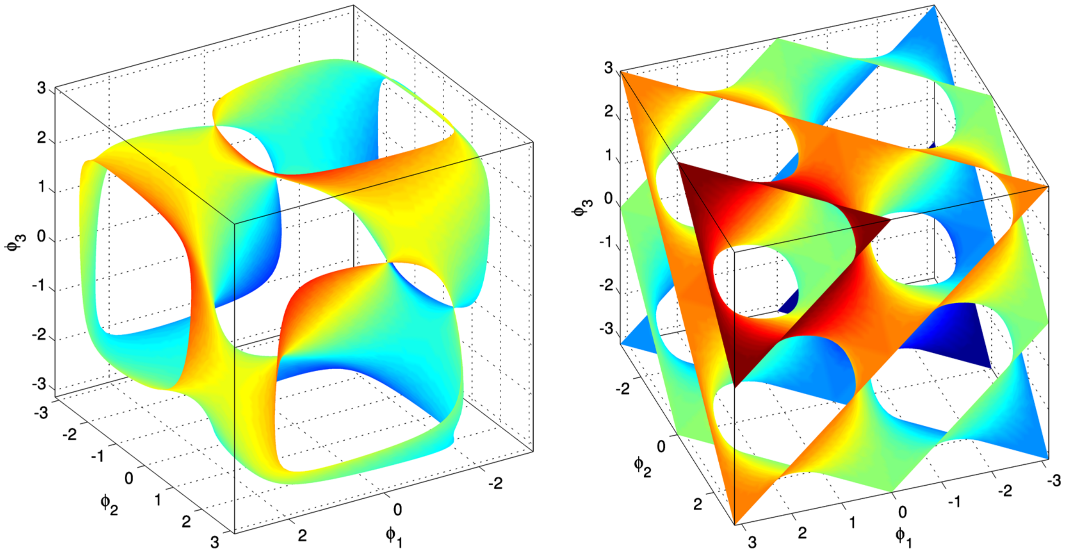


Fig. 3 Singularities of the trident snake with active joints (*left*) and active wheels (*right*)

$|\det\Gamma(q)| > \epsilon_d$, $\epsilon_d > 0$ and $\phi_{\min i} \le \phi_i \le \phi_{\max i}$ for $i = 1, 2, 3$. In order to take them into account, we need to extend the system (18) by adding an auxiliary state variable q_{n+1} and an output variable $y_{p+1} = q_{n+1}$. The equations of the extended system will assume the following form

$$\begin{cases} \dot{q} = G(q)\eta, \\ \dot{q}_{n+1} = p(\epsilon_d + \det\Gamma(q), \alpha) \\ \qquad\qquad + \sum_{i=1}^{3}(p(\phi_{\min i} - \phi_i, \alpha) + p(\phi_i - \phi_{\max i}, \alpha)), \\ \dot{\eta} = v, \quad y_{\text{ext}} = (h(q,\eta), q_{n+1}). \end{cases} \tag{19}$$

In the extended system, the function $p(x, \alpha) = x + \frac{1}{\alpha}\ln(1 + \exp(-\alpha x))$, $\alpha > 0$, represents a smooth approximation of the function $\max\{x, 0\}$, and the invertibility condition has been implemented in the form $\det\Gamma(q) \le -\epsilon_d$, which corresponds to $\det\Gamma(q_0) < 0$. The case of $\det\Gamma(q_0) > 0$ is handled in a similar way.

Alternatively to Eq. 19, two auxiliary state variables can be introduced, responsible separately for the invertibility condition and collision avoidance, in which case the extended system is

$$\begin{cases} \dot{q} = G(q)\eta, \\ \dot{q}_{n+1} = p(\epsilon_d + \det\Gamma(q), \alpha), \\ \dot{q}_{n+2} = \sum_{i=1}^{3}(p(\phi_{\min i} - \phi_i, \alpha) + p(\phi_i - \phi_{\max i}, \alpha)), \\ \dot{\eta} = v, \quad y_{\text{ext}} = (h(q,\eta), q_{n+1}, q_{n+2}). \end{cases} \tag{20}$$

The motion planning problem in the extended system consists of defining a control $u(t)$, such that $y_{\text{ext}}(T) = y_{\text{ext d}} = (y_d, 0)$ in the first case or $y_{\text{ext}}(T) = y_{\text{ext d}} = (y_d, 0, 0)$ in the second case.

3.1 Singularity Robust Jacobian Inverse

In order to solve the motion planning problem we may apply the Jacobian pseudo-inverse algorithm [13]. However, it turns out that when the constraints are satisfied, the Jacobian of the extended system becomes singular [15]. There are two ways of circumventing this difficulty. First, we can replace the Jacobian pseudo-inverse by the singularity robust Jacobian inverse. In accordance to this method, we need to determine a 1-parameter family of controls $u_\vartheta(t)$, depending on $\vartheta \in R$, that solves the functional differential equation

$$\frac{du_\vartheta(t)}{d\vartheta} = -\gamma C_\vartheta^T(T) B_\vartheta^T(t) \Phi_\vartheta^T(T,t) \times (\mathcal{D}(u_\vartheta(\cdot)) + \kappa I_{p+\nu})^{-1}(y_{\text{ext}}(T) - y_{\text{ext d}}), \tag{21}$$

$\nu = 1, 2$ for, respectively, systems (19), (20), and then to compute a solution $u_d(t)$ of the motion planning problem by passing to the limit $u_d(t) = \lim_{\vartheta\to+\infty} u_\vartheta(t)$. The parameter $\gamma > 0$ defines the speed of convergence of the motion planning algorithm, while $\kappa > 0$ provides a regularization of the matrix $\mathcal{D}(u_\vartheta(\cdot))$.

The matrix functions $B_\vartheta(t)$, $C_\vartheta(t)$, $\Phi_\vartheta(t,s)$ and $\mathcal{D}(u_\vartheta(\cdot))$ appearing on the right hand side of Eq. 21 refer to the linear approximation of the system (19) along the trajectory $(q_\vartheta(t), q_{n+1\,\vartheta}(t), \eta_\vartheta(t))$ corresponding to the control $u_\vartheta(t)$, Tchoń and Jakubiak [13]. For the class of control affine systems

$$\dot{x} = f(x) + F(x)u, \quad y = k(x) \tag{22}$$

in R^w, that includes Eqs. 19 and 20, these functions are defined as follows: Given a control $u(t)$ and an initial state x_0, let $x(t)$ denote the corresponding trajectory of the control affine system. The linear approximation of this system is a linear, time-variant control system of the form

$$\dot{\xi} = A(t)\xi + B(t)v, \quad \zeta = C(t)\xi,$$

where

$$A(t) = \frac{\partial(f(x(t)) + F(x(t))u(t))}{\partial x},$$

$$B(t) = F(x(t)),$$

$$C(t) = \frac{\partial k(x(t))}{\partial x},$$

$$\mathcal{D}(u(\cdot)) = C(T)\int_0^T \Phi(T,s)B(s)B^T(s) \times \Phi^T(T,s)ds\,C^T(T),$$

and the matrix $\Phi(t,s)$ comes as the solution to the evolution equation $\frac{\partial\Phi(t,s)}{\partial t} = A(t)\Phi(t,s)$, with initial condition $\Phi(s,s) = I_w$. The function

$$J_{x_0,T}(u(\cdot))v(\cdot) = \zeta(T) = C(T)\int_0^T \Phi(T,t)B(t)v(t)dt$$

is called the Jacobian of the system (22). Various inverses of the Jacobian give rise to various Jacobian motion planning algorithms [13].

3.2 Imbalanced Jacobian Inverse

Another way to prevent the Jacobian pseudo-inverse algorithm from getting ill-conditioned, relies on regularization of the extended systems by adding a quadratic function $\rho(q) = \frac{1}{2}\phi^T S\phi$ of the joint angles $\phi = (\phi_1, \phi_2, \phi_3)^T$, where S denotes a matrix. The so regularized system (19) assumes the form

$$\begin{cases} \dot{q} = G(q)\eta, \\ \dot{q}_{n+1} = p(\epsilon_d - \det\Gamma(q), \alpha) \\ \qquad + \sum_{i=1}^{3}(p(\phi_{\min i} - \phi_i, \alpha) \\ \qquad + p(\phi_i - \phi_{\max i}, \alpha)) + \rho(q), \\ \dot{\eta} = v, \quad y_{\text{reg}} = (h(q,\eta), q_{n+1}). \end{cases} \tag{23}$$

If two state variables have been added, two independent quadratic functions $\rho_1(q)$, $\rho_2(q)$ need to be used, for example

$$\begin{cases} \dot{q} = G(q)\eta, \\ \dot{q}_{n+1} = p(\epsilon_d + \det\Gamma(q), \alpha) + \rho_1(q), \\ \dot{q}_{n+2} = \sum_{i=1}^{3}(p(\phi_{\min i} - \phi_i, \alpha) \\ \qquad + p(\phi_i - \phi_{\max i}, \alpha)) + \rho_2(q), \\ \dot{\eta} = v, \quad y_{\text{reg}} = (h(q,\eta), q_{n+1}, q_{n+2}). \end{cases} \tag{24}$$

Observe that, since the regularized systems contain the $\rho(q)$ functions, it is not possible to assign a desired value to y_{reg}. This problem is overcome by the imbalanced Jacobian whose main idea rests on applying the Jacobian pseudo-inverse algorithm to the dynamics of the regularized system fed up by the error coming from the extended system. Conditions under which such a procedure is justified have been stated in [15]. The resulting imbalanced Jacobian algorithm is determined by the following functional differential equation

$$\frac{du_\vartheta(t)}{d\vartheta} = -\gamma C^T_{\text{reg}\vartheta}(T)B^T_{\text{reg}\vartheta}(t)\Phi^T_{\text{reg}\vartheta}(T,t) \times \mathcal{D}^{-1}_{\text{reg}}(u_\vartheta(\cdot))(y_{\text{ext}}(T) - y_{\text{ext d}}). \tag{25}$$

All the data subscripted with "reg" appearing in Eq. 25 need to be computed for the control $u(t)$ applied to the regularized system (23) or (24), while the output error in Eq. 25 comes from either Eq. 19 or Eq. 20. As before, a solution to the motion planning problem is obtained as the limit $u_d(t) = \lim_{\vartheta\to+\infty} u_\vartheta(t)$.

3.3 Numeric Computations

Usually, in order to make the computations efficient, the systems (21) or (25) are transformed to a discrete and parametric form, using e.g. the Euler integration scheme and a finite-dimensional representation of controls, like by the truncated harmonic series

$$u(t) = u_\lambda(t) = \lambda_0 + \sum_{i=1}^{s}(\lambda_{2i-1}\sin i\omega t + \lambda_{2i}\cos i\omega t), \tag{26}$$

where the frequency $\omega = \frac{2\Pi}{T}$. The equation (26) can be represented in the matrix form

$$u_\lambda(t) = P(t)\lambda,$$

$$P(t) = \begin{bmatrix} 1 & \sin\omega t & \cos\omega t & \dots & \cos s\omega t & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 1 & \sin\omega t & \dots & 0 \\ \vdots & & & & & & & & \vdots \\ 0 & \dots & & & & \dots & 0 & \dots & \cos s\omega t \end{bmatrix}.$$

Given the control function $u_\lambda(t)$, the matrix functions defining the linear approximation of Eq. 22 will be denoted as $A_\lambda(t)$, $B_\lambda(t)$, $C_\lambda(t)$, $\Phi_\lambda(t)$, robot's trajectory $x(t) = x_\lambda(t)$, and the tracking error $e(\lambda)$. In order to update the control functions, the control coefficients λ are modified according to the following rule

$$\lambda_{\vartheta+1} = \lambda_\vartheta - \gamma J^{\#}_{x_0,T}(\lambda_\vartheta)e(\lambda_\vartheta), \qquad \vartheta = 0, 1, \ldots, \tag{27}$$

starting from an initial λ_0. In Eq. 27, the Jacobian inverse

$$J^{\#}_{x_0,T}(\lambda) = J^{T}_{x_0,T}(\lambda)\left(J_{x_0,T}(\lambda)J^{T}_{x_0,T}(\lambda) + \kappa I_{p+\nu}\right)^{-1},$$

$\nu = 1$ or 2, in the case of the singularity robust Jacobian pseudo-inverse (21) or

$$J^{\#}_{x_0,T}(\lambda) = J^{T}_{x_0,T}(\lambda)\left(J_{x_0,T}(\lambda)J^{T}_{x_0,T}(\lambda)\right)^{-1}$$

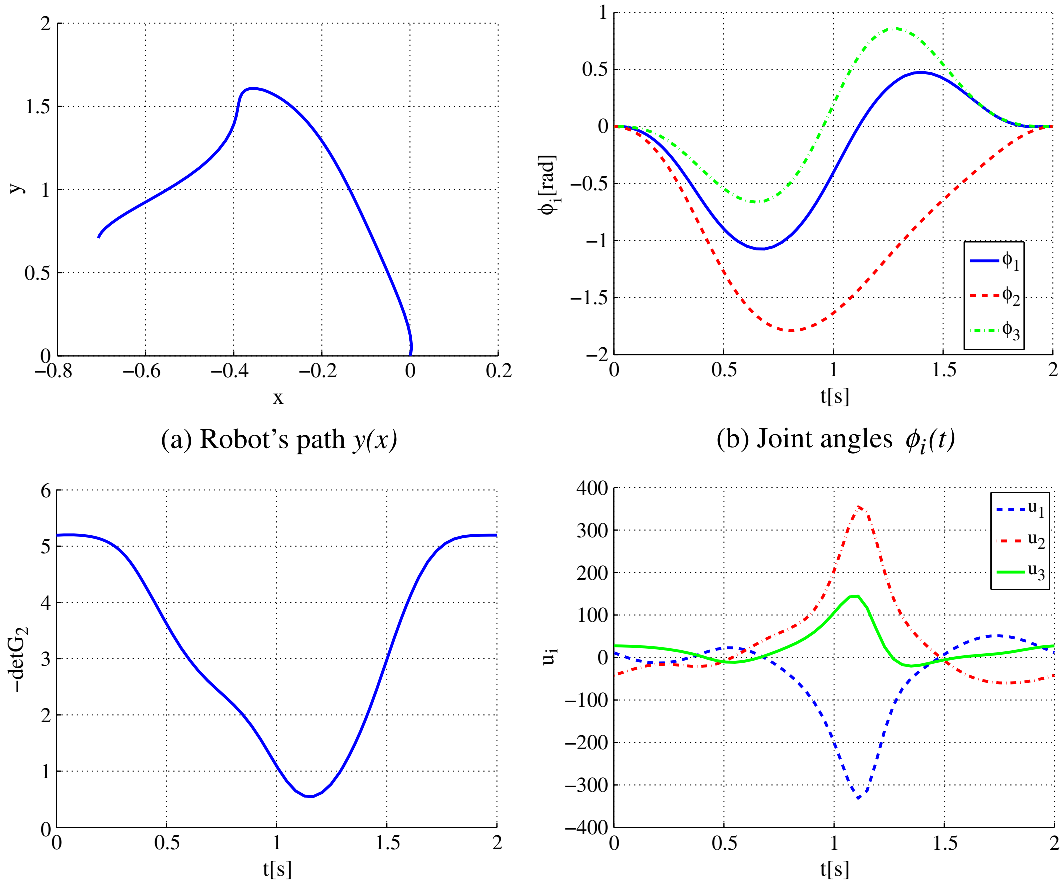
for the imbalanced Jacobian algorithm (25). The Jacobian matrix

$$J_{x_0,T}(\lambda) = C_\lambda(T)\int_0^T \Phi_\lambda(T,t)B_\lambda(t)P(t)dt$$

uses, respectively, the data from either the extended or the regularized system. The output error $e(\lambda)$ corresponds to the extended system.

4 Computer Simulations

For performance evaluation of the proposed motion planning algorithms, appropriate computer simulations have been executed. For the model of the trident snake equipped with passive wheels



(a) Robot's path *y(x)*

(b) Joint angles $\phi_i(t)$

(c) Plot of $-detG_2(t)$

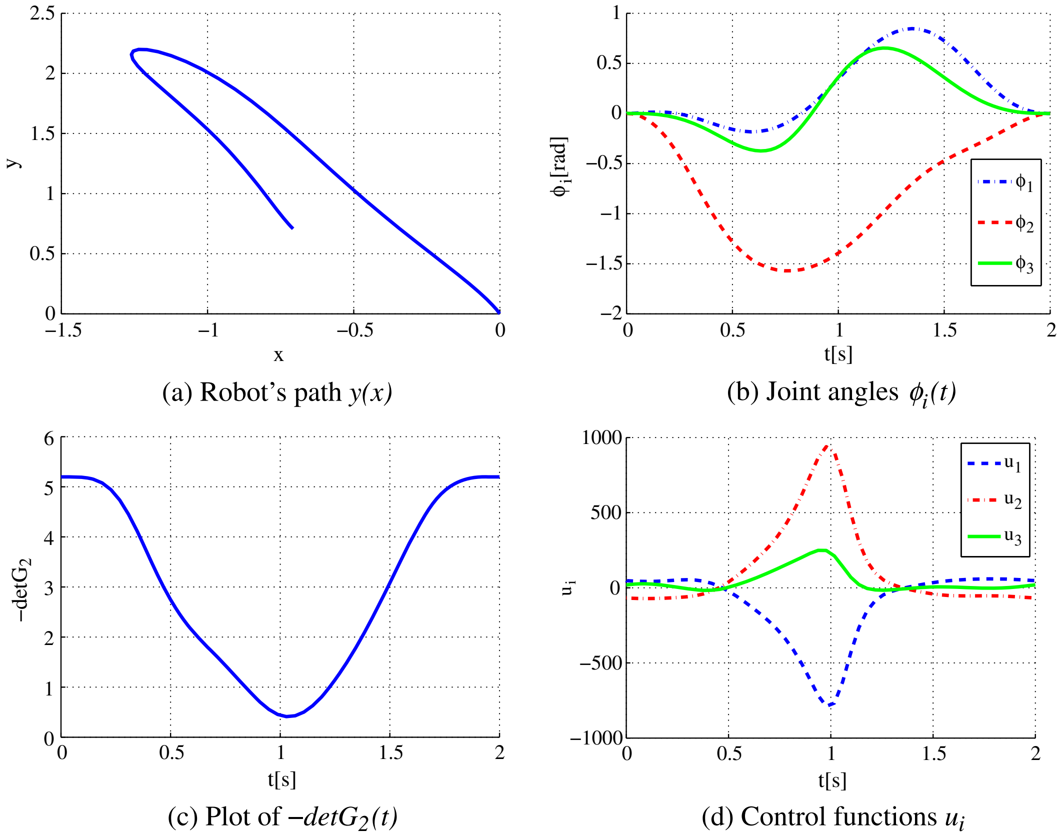
(d) Control functions u_i

Fig. 4 Active joints: singularity robust Jacobian algorithm

two control algorithms have been assessed: the singularity robust Jacobian algorithm and the imbalanced Jacobian algorithm. The motion planning problem chosen for the robot equipped with active wheels has been solved by applying only the former algorithm. As described in previous sections, execution of the task requires taking into account constraints imposed on the movement of the trident snake, concerning both the determinant of matrix G_2 from Eq. 7 or T from Eq. 14, and on the joint angles' ranges allowing a collision-free motion, thus one or two additional state variables have been introduced. All motion planning algorithms have been implemented within the Matlab/Simulink programming environment. Measure units used on the charts presented in the following subsections are: 1 dm, 1 s, 1 dag and 10^{-3} Nm.

4.1 Trident Snake with Passive Wheels and Active Joints

Since the dynamics model of the robot with passive wheels presented in Eq. 2 contains certain simplifications, the mass and geometry parameters of the trident snake have been chosen somewhat arbitrarily and are respectively: $M_0 = 10, M_k = 1, l = 1, r = 1, R_k = 0.1, d = 0.01$. Possible variability of the determinant of matrix G_2 has been restricted by requiring $\det G_2 \leq -0.1$ to avoid singularities. In order to ensure collision-free motion, the inequalities $-\frac{2\pi}{3} \leq \phi_i \leq \frac{2\pi}{3}$ have been imposed set for $i = 1, 2, 3$. The remaining inputs for the simulations are $T = 2$, $\gamma = 0.5$, $\epsilon = 0.001$ (required motion planning accuracy), $\alpha = 90$, $\kappa = 0.01$. The problem is to determine control coefficients $\lambda \in R^{12}$ in Eq. 26, for which



(a) Robot's path *y(x)*

(b) Joint angles $\phi_i(t)$

(c) Plot of $-detG_2(t)$

(d) Control functions u_i

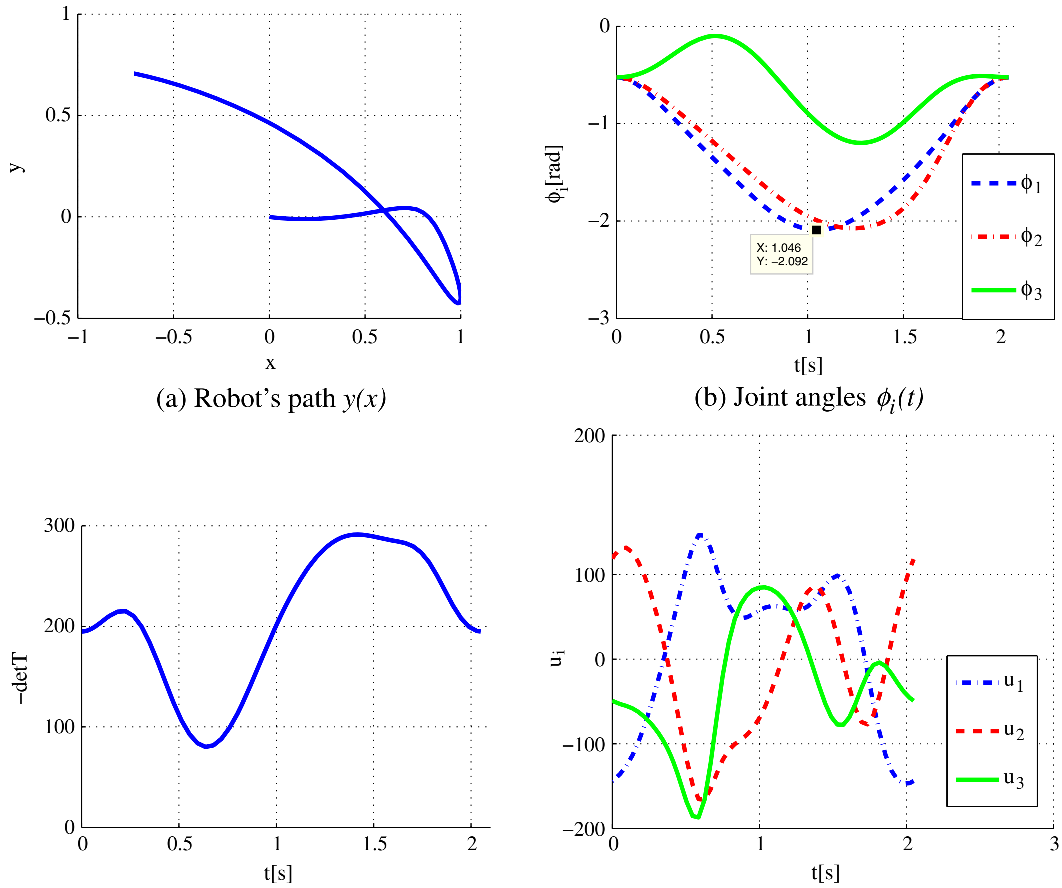
Fig. 5 Active joints: imbalanced Jacobian algorithm

the robot will move from $(x_0, y_0, \theta_0, \phi_{10}, \phi_{20}, \phi_{30}, \eta_{10}, \eta_{20}, \eta_{30})^T = \left(-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, \frac{\pi}{16}, 0, 0, 0, 0, 0, 0\right)^T$ to $(x_d, y_d, \theta_d, \phi_{1d}, \phi_{2d}, \phi_{3d}, \eta_{1d}, \eta_{2d}, \eta_{3d})^T = 0$, and stop in the final posture. Initial control parameters are $\lambda_0 = (0.5, 0.5, 0.5, 0.5, -0.5, 0.5, 0.5, 0.5, -0.3, 0.3, 0.3, 0.3)^T$.

The singularity robust Jacobian algorithm solved the problem in 47 iterations. Results' summary can be found in Fig. 4, usage of two extra state variables instead of one in this case had no influence on the observed behaviour. The imbalanced Jacobian algorithm found the solution shown in Fig. 5 in 68 iterations. Two additional state variables have been used, and additional robustness have been required. Quadratic functions used: $\rho_1 = \phi_1^2 + \phi_2^2 + \phi_3^2$, $\rho_2 = 2\phi_1^2 + \phi_2^2 + 3\phi_3^2$.

4.2 Trident Snake with Active Wheels and Passive Joints

The dynamics model for the trident snake with active wheels reflects physical parameters of the robot shown in Eq. 1. These parameters are: $M_0 = 52$, $M_k = 3$, $R_k = 0.2$, $M_l = 15$, $M_s = 5.5$, $r = 1.2$, $l = 1.1$, $d = 0.01$. The charts (6) show simulation results for the singularity robust Jacobian algorithm with one extra state variable, complying with the data $T = 2.05$, $\gamma = 0.5$, $\epsilon = 0.001$, $\alpha = 90$, $\epsilon_d = 10^{-1}$, $\kappa = 0.01$, $\phi_{\min i} = -\frac{2\pi}{3}$, $\phi_{\max i} = \frac{2\pi}{3}$, $i = 1, 2, 3$. The initial conditions, the goal of the motion, and the initial control parameters have been selected, respectively, as follows: $(x_0, y_0, \theta_0, \phi_{10}, \phi_{20}, \phi_{30}, \beta_{10}, \beta_{20},$



(a) Robot's path $y(x)$

(b) Joint angles $\phi_i(t)$

(c) Plot of $-detT(t)$

(d) Control functions u_i

Fig. 6 Active wheels: singularity robust Jacobian algorithm

$\beta_{30}, \eta_{10}, \eta_{20}, \eta_{30})^T = \left(-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, 0, -\frac{\pi}{6}, -\frac{\pi}{6}, -\frac{\pi}{6}, 0, 0, 0, 0, 0, 0\right)^T$, $(x_d, y_d, \theta_d, \phi_{1d}, \phi_{2d}, \phi_{3d}, \eta_{1d}, \eta_{2d}, \eta_{3d})^T = (0, 0, 0, -\frac{\pi}{6}, -\frac{\pi}{6}, -\frac{\pi}{6}, 0, 0, 0)^T$, $\lambda_0 = (0.8, 0.7, 0.7, 0.7, 0.8, 0.8, 0.7, 0.7, 0.7, 0.8, 0.8, 0.7, 0.7, 0.7, 0.8)^T$. To prevent the determinant $|\det T|$ from getting very large, in computations the value of $\det T$ has been replaced with $-1 - \epsilon_d$, when the condition $\det T \leq -\epsilon_d$ is satisfied [8]. The requested accuracy of the solution has been achieved after 787 iterations.

5 Conclusion

Using the endogenous configuration approach we have designed constrained Jacobian motion planning algorithms for the dynamic model of the trident snake robot with either active joints and passive wheels or passive joints and active wheels. Performance of these algorithms has been tested by computer simulations. It may be concluded that the endogenous configuration space approach has passed this test. Future research should be devoted to experimental verification of these algorithms as well as will address the motion planning problem in more complex structures of the robot (multi-link arms). Taking into account increasing complexity of motion planning for the trident snake with growing number of links, the motion planning problem for the trident snake robot can be regarded as a kind of benchmark problem for motion planning algorithms.

Acknowledgement The authors are very much indebted to anonymous reviewers whose criticism has helped them to improve the contents and the presentation of this paper.

References

1. Hirose, S.: Biologically Inspired Robots—Snake-like Locomotors and Manipulators. Oxford University Press, Oxford (1993)
2. Transeth, A.A., Pettersen, K.Y., Liljebäck, P.: A survey on snake robot modeling and locomotion. Robotica **27**, 999–1015 (2009)
3. Ishikawa, M.: Trident snake robot: locomotion analysis and control. In: Proc. IFAC NOLCOS, pp. 1169–1174. Stuttgart, Germany (2004)
4. Ishikawa, M., Minami, Y., Sugie, T.: Development and control experiment of the trident snake robot. In: Proc. 45th IEEE CDC, pp. 6450–6455. San Diego, CA (2006)
5. Ishikawa, M., Minami, Y., Sugie, T.: Development and control experiment of the trident snake robot. IEEE/ASME Trans. Mechatronics **15**, 9–15 (2010)
6. Ishikawa, M., Fujiro, T.: Control of the double-linked trident snake robot based on the analysis of its oscillatory dynamics. In: Proc. 2009 IEEE/RSJ IROS, pp. 1314–1319. St. Louis, MI (2009)
7. Jakubiak, J., Tchoń, K., Janiak, M.: Motion planning of the trident snake robot: an endogenous configuration space approach. In: Robot Design, Dynamics, and Control, pp. 159–166. Springer-Verlag, Wien, New York (2010)
8. Paszuk, D., Tchoń, K., Pietrowska, Z.: Motion planning of the trident snake robot equipped with passive or active wheels. In: Bull. Polish Academy of Sciences, ser. Technical Sciences, vol. 60, pp. 547–554 (2012)
9. Tchoń, K., Jakubiak, J.: Motion planning of the double-link trident snake robot. In: Proc. 10th IFAC SYROCO, pp. 337–342. Dubrovnik, Croatia (2012)
10. Ishikawa, M., Morin, P., Samson, C.: Tracking control of the trident snake robot with the transverse function approach. In: Proc. 48th IEEE CDC, pp. 4137–4143. Shanghai, China (2009)
11. Magiera, W.: Non-symmetric trident snake: the transverse function approach. In: Scientific Papers of Warsaw University of Technology, vol. 182, pp. 451–460 (in Polish) (2012)
12. Gospodarek, Sz.: Design and modeling of a a trident snake mobile robot. MSc Thesis, Wrocław University of Technology (in Polish) (2011)
13. Tchoń, K., Jakubiak, J.: Endogenous configuration space approach to mobile manipulators: a derivation and performance assessment of Jacobian inverse kinematics algorithms. Int. J. Control **76**, 1387–1419 (2003)
14. Tchoń, K., Jakubiak, J., Małek, Ł.: Motion planning of nonholonomic systems with dynamics. In: Computational Kinematics, pp. 125–132. Springer-Verlag, Berlin (2009)
15. Janiak, M., Tchoń, K.: Constrained motion planning of nonholonomic systems. Syst. Control Lett. **60**, 625–631 (2011)
16. Yamaguchi, H.: Dynamical analysis of an undulatory wheeled locomotor: a trident steering walker. In: Proc. 10th IFAC SYROCO, pp. 157–164. Dubrovnik, Croatia (2012)
17. Pietrowska, Z.: Kinematics, dynamics and control of a trident snake non-holonomic system. MSc Thesis, Wrocław University of Technology (in Polish) (2012)
18. Lewis, A.: When is a mechanical control system kinematic? In: Proc. 38th IEEE CDC, pp. 1162–1167. Phoenix, AZ (1999)