Modeling and Control of a Snake-like robot on a smooth surface

Takeru Yanagida¹,* and Masami Iwase²

1Department of Advanced Multidisciplinary Engineering, Graduate School of Advanced Science and Technology, Tokyo Denki University, 5 Senjuasahi-cho, Adachi-ku, Tokyo 120-8551, Japan
2Department of Robotics and Mechatronics, School of Science and Technology for Future Life, Tokyo Denki University, 5 Senjuasahi-cho, Adachi-ku, Tokyo 120-8551, Japan

¹yanagida, iwase@ctrl.fr.dendai.ac.jp

Abstract. In this paper, motion control of a snake-like robot for winding and moving on smooth surface is developed. To design the motion control, a mathematical model of the snake-like robot is derived by Projection method. The Projection method leads both the mathematical model and constraints working in the mathematical model. The constraints give important information for winding and moving on the surface. State-Dependent Riccati Equation (SDRE) control strategy taking the constraints into account realizes winding and moving on the surface for the snake-like robot. The effectiveness of the proposed method is verified through numerical simulations.

Keywords: Snake-like Robot, Modeling, Motion Control, Computer simulation

1. Introduction

Autonomous robots are expected to work for critical tasks including rescue activities, architecture maintenance, environment keeping and so on. The ability of adaptation to the environment surround such a robot is important because the robot should move in the environment for working.

Living snakes have quite superior ability with respect to locomotion, and can move even on rough ground and climb trees by winding action including undulating and bending. Hence robotic researcher are interested in living snakes’ high mobility capability. This fact motivates these researchers to develop snake-like robots which are expected to have high mobility capability as same as living snakes [1, 2, 3, 4, 5, 6, 7].

Locomotion control for snake-like robot has been studies mainly in two-dimensional space so far. However, locomotion control of snake-like robots in three-dimensional space becomes much attractive after 2005. For example, a winding movement for moving on
a smooth surface by torsional movement is proposed by [8, 9]. These studies have a big impact and realize the interesting movement. On the other hand, the proposed torsional movement for climbing-up has a big problem that such a snake-like robot get stack and finally cannot climb a tree up if there are blockades like branches on the tree.

Therefore, in this study, we aim to propose a completely different locomotion control for snake-like robots to moving on a surface with arbitrary curvature by winding motion. To realize the locomotion control, we extend our previous two works; the winding motion control for snake-like robots around a column [11], and State Dependent Riccati Equation (SDRE) control strategy for snake-like robots moving on a flat plane studied by [10].

The two previous works use a two-dimensional mathematical model of the snake-like robot, and suppose that the ground shape is simple, i.e. flat plane and slope. It implies that the two-dimensional snake-like robot model cannot be utilized for analysis and control design of winding and moving on the surface. Hence, we derive a three-dimensional model of snake-like robots, by Projection Method [12, 13], under the situation that the ground shape is defined by an arbitrary curvature smooth function.

Based on the derived three-dimensional model of snake-like robots, the SDRE control strategy is designed, and applied to the model for obtaining a controller to make the robot winding and moving on the surface. The effectiveness of the proposed modeling and control strategy is verified through numerical simulations.

2. Modeling of snake-like robot

The snake-like robot in this paper consists of multi modules. Each module has thin disk shape with a rod penetrating the disk center as shown in Figure 1. Each rod is connected to other one in serial via a joint. An actuator with two degrees of freedom in yaw and pitch axes is installed to each joint as shown in Figure 1. This configuration can be extended to general model with \( n \)-links snake-like robot.

The equations of motion of the robot is derived by the Projection Method. The Projection Method is a modeling method. The method derives an equation of motion as following process; (1) derive every constraint-free dynamics model of each component consisting of a system, (2) derive every constraints working among components and environment, (3) derive the combined equations of motion by combining the constraint-free models and constraints. The equations of motion for the snake-like robot in this paper is derived along the process.

The important point for the Projection method is how to derive the constraints. For the snake-like robot case, we start to consider constraints working for the disk moving with non side-slip on a smooth surface with arbitrary curvature. After that, the constraints working for among the related modules. Considering the all constraints, the equations of motion for the snake-like robot are derived by the projection operation.

2.1. Modeling of module moving on smooth surface with arbitrary curvature

The constraints working for the modules moving with non side-slipping on smooth surface with arbitrary curvature are derived. The smooth surface can be defined as a smooth shape function which is differentiable at least one time everywhere.
2.1.1. Coordinate system

The coordinates system for deriving the equations of motion is defined here. The initial position and attitude of a module is set as the longer direction of the rod penetrating the disk center is equal to the $y$-axis in the global coordinate system. Similarly the $x$- and $z$- axes of the module is set equal to the $x$- and $z$- axes in the global coordinate. The module attitude is indicated by Z-X-Y Euler angles. The Z-X-Y Euler angles are represented $\phi, \theta, \psi$ as defined in Figure 2.

\[
\dot{x} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix},
\]

where $(x, y, z)$ represents the position of the center of gravity of the module.

A new velocity coordinate but related to (1) is introduced. It is called 'pseudo velocity coordinate'. The pseudo velocity coordinate is suitable for defining constraints and representing the module motion:

\[
v_{tr} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}.
\]

The coordinate transform matrix relating the global coordinate to the pseudo velocity is defined by a rotation matrix:

\[
\dot{x} = A_v v_{tr},
\]

where $A_v := R_z(\phi) R_x(\theta)$, $R_z(\phi) := \begin{bmatrix} \cos \phi & -\sin \phi & 0 \\ \sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $R_x(\theta) := \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix}$.

The matrix $A_v$ is described by two Euler angles; $\phi$ and $\theta$.

The rotational motion of the module is represented by the angular velocity defined with the principal axes of inertia of the module. The angular velocity with the principal axes of inertia is defined as

\[
\omega_I = \begin{bmatrix} \omega_{Ix} \\ \omega_{Iy} \\ \omega_{Iz} \end{bmatrix}^T.
\]
The pseudo angular velocity coordinates is defined in the similar way. We should obtain the consistency between the pseudo velocity in the translational direction and the pseudo angular velocity. To do so, the angular velocity with the principal axes of inertia should be rotated around $\psi$ axis in Figure 2. As the result, the pseudo angular velocity coordinate can be represented as

$$\omega = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T.$$

The coordinate transform matrix between the angular velocity with the principal axes of inertia and the pseudo angular velocity is defined by a rotation matrix:

$$\omega_f = A_\omega \omega \quad (3)$$

$$A_\omega := R_y(\psi)$$

$$R_y(\psi) := \begin{bmatrix} \cos \psi & 0 & \sin \psi \\ 0 & 1 & 0 \\ -\sin \psi & 0 & \cos \psi \end{bmatrix}$$

Finally, the coordinate transform matrix between the Euler angular velocity and the pseudo angular velocity is defined by considering the Euler rotation as shown in Figure 3:

$$\omega = A_\phi \Phi \quad (4)$$

$$A_\phi := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \sin \theta \\ 0 & 0 & \cos \theta \end{bmatrix}$$

where $\Phi = [\theta \quad \psi \quad \phi]^T$.

### 2.1.2. Constraint-free module’s equations of motion

The equations of motion of a module in the translational motion is represented as

$$\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}$$

$$\Leftrightarrow \dot{M}_f \ddot{x} = h_f \quad (5)$$

where

$$\dot{M}_f := \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix}, \quad h_f := \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}.$$
The equations of motion of a module in the rotation motion with the principal axes of inertia is represented as

\[
\begin{bmatrix}
  j_x & 0 & 0 \\
  0 & j_y & 0 \\
  0 & 0 & j_z
\end{bmatrix}
\begin{bmatrix}
  \dot{\omega}_{lx} \\
  \dot{\omega}_{ly} \\
  \dot{\omega}_{lz}
\end{bmatrix}
= \begin{bmatrix}
  (j_y - j_z) \omega_{ly} \omega_{lz} + \tau_x(i) - \tau_x(i+1) \\
  (j_z - j_x) \omega_{lz} \omega_{lx} \\
  (j_x - j_y) \omega_{lx} \omega_{ly} + \tau_z(i) - \tau_z(i+1)
\end{bmatrix}
\]

\[
\Leftrightarrow \mathbf{J}_f \dot{\omega}_f = n_f(\omega_f)
\tag{7}
\]

where

\[
\mathbf{J}_f := \begin{bmatrix}
  j_x & 0 & 0 \\
  0 & j_y & 0 \\
  0 & 0 & j_z
\end{bmatrix}, \quad n_f(\omega_f) := \begin{bmatrix}
  (j_y - j_z) \omega_{ly} \omega_{lz} + \tau_x(i) - \tau_x(i+1) \\
  (j_z - j_x) \omega_{lz} \omega_{lx} \\
  (j_x - j_y) \omega_{lx} \omega_{ly} + \tau_z(i) - \tau_z(i+1)
\end{bmatrix},
\]

where \(\tau_x, \tau_z\) are input torque generated by an actuator installed into a joint. \(i\) means the index indicating \(i\)-th link \((i = 1, 2, \ldots, n)\). Rearranging (7) by (3) leads to

\[
\mathbf{A}_w^T \mathbf{J}_f \mathbf{A}_w \dot{\omega} = \mathbf{A}_w^T n_f(\mathbf{A}_w \omega) - \mathbf{A}_w^T \mathbf{J}_f \dot{\mathbf{A}}_w \omega
\]

\[
\Leftrightarrow \mathbf{J} \dot{\omega} = \mathbf{n}
\tag{8}
\]

where

\[
\mathbf{J} := \mathbf{A}_w^T \mathbf{J}_f \mathbf{A}_w, \quad \mathbf{n} := \mathbf{A}_w^T n_f(\mathbf{A}_w \omega) - \mathbf{A}_w^T \mathbf{J}_f \dot{\mathbf{A}}_w \omega.
\]

As the results, (7) and (8) are the constraints-free equations of motion of a module.
2.1.3. Constraints working for modules

The constraints play an important role in modeling processes. Here, we derive the constraints working for a module moving with non side-slip on a smooth surface with arbitrary curvature. We support that

- A module moves on a smooth surface.
- A module moves with non side-slip.

The smooth surface is given by a shape function \( f_y = 0 \). We also suppose the shape function is differentiable at least one time everywhere. The normal vector of the shape function at a point is defined as

\[
\mathbf{n}_g := \frac{\partial f_y}{\partial \mathbf{x}_g},
\]

\[
\mathbf{x}_g = [x_g, y_g, z_g]^T
\]

where \( \mathbf{x}_g \) is the state vector to define the surface shape in Figure 4.

![Figure 4: Sample of shape function and its normal vector](image)

The constraints of a module moving on the surface indicate to keep the geometrical relation between the center of gravity of the module and the contact point where the module touches with the surface to be constant. Thus, the geometrical constraints are represented as

\[
\Phi_1 : x - x_r - x_g = 0, \quad \Phi_2 : y - y_r - y_g = 0, \quad \Phi_3 : z - z_r - z_g = 0,
\]
where \((x_r, y_r, z_r)\) is the local coordinate in the module coordinate system, and indicates the relation between the center of gravity and the contact point of the module. The relationship among the center of gravity, the contact point and the local coordinate is represented by the following simultaneous equations;

\[
\begin{align*}
\dot{x}^2 + y^2 + z^2 &= r^2 \\
\mathbf{n}_g \cdot [x_r, y_r, z_r]^T &= 0 \\
\mathbf{n}_g \times \mathbf{n}_p / (||\mathbf{n}_g|| ||\mathbf{n}_p||) \cdot [x_r, y_r, z_r]^T &= 0 \\
\mathbf{n}_p := \begin{bmatrix}
-\cos \theta \sin \phi \\
\cos \theta \cos \phi \\
\sin \theta
\end{bmatrix},
\end{align*}
\]

where \(\mathbf{n}_p\) is a unit vector representing the longer direction of the module in Figure 5. The operator, “\(\cdot\)”, in the above equations indicates inner product. The operator, “\(\times\)”, means cross product.

![Figure 5: Geometrical constraints of \(\Phi_1, \Phi_2\) and \(\Phi_3\)](image)

The contact point of a module should stay on the surface. In other words, we suppose that the module never leave from the surface. To realize the constraint, we introduce a virtual
mass on the surface and its dynamics as
\[
\begin{bmatrix}
m_g & 0 & 0 \\
0 & m_g & 0 \\
0 & 0 & m_g
\end{bmatrix}
\begin{bmatrix}
x_g \\
y_g \\
z_g
\end{bmatrix} = \begin{bmatrix} 0 \\
0 \\
0
\end{bmatrix}
\]
\[\iff M_g \ddot{x}_g = h_g \tag{12}\]

where
\[
M_g := \begin{bmatrix}
m_g & 0 & 0 \\
0 & m_g & 0 \\
0 & 0 & m_g
\end{bmatrix}, \quad h_g := \begin{bmatrix} 0 \\
0 \\
0
\end{bmatrix}.
\]

\(m_g\) is virtual mass, which will be set zero after deriving the equations of motion. In other words, in the modeling process, we suppose the contact point has a nonzero virtual mass \(m_g\).

The contact point should stay on the surface. It means that the contact point mass does NOT have a velocity in the normal direction of the surface at the contact point. That is, the constraints can be represented as
\[
\Phi_4 : n_{g} \cdot \begin{bmatrix} \dot{x}_g \\ \dot{y}_g \\ \dot{z}_g \end{bmatrix} = 0 \tag{13}
\]

The module moves on the surface with non side-slip. It means the translational velocity of the center of gravity is related to the peripheral speed at the contact point as shown in Figure 6. Thus, the condition can be written by
\[
\Phi_5 : (n_p \times n_g) \cdot \begin{pmatrix} R_z(\phi)R_x(\theta) \cdot \begin{bmatrix} v_x \\ 0 \\ v_z \end{bmatrix} \end{pmatrix} - r\omega_y = 0 \tag{14}
\]

and
\[
n_a := (x - x_g)/||x - x_g||.
\]

### 2.2. Constraints working for between modules

We derive the constraints working for between modules connecting. We support that

- Each modules are connected by serial-connected link.
- Joint is constraint-velocity joint.

Distinguish using the subscript of i a module of multiple. The top of the module subscript number is 1, the second module subscript number is 2.

The condition that the each modules connecting by serial-connected link(Figure 7) is represented as
\[
\Phi_6 : x_{i+1} + l_{i+1}n_{p(i+1)} + l_{i}n_{p(i)} - x_i = 0 \quad (\in \mathbb{R}^{3 \times 1}). \tag{15}
\]

Connecting by constant-velocity joint means that the each module rotation of the axis is same amount(Figure 8). Thus, the condition is defined as
\[
\Phi_7 : \psi_{i+1} - \psi(i) = 0. \tag{16}
\]
2.3. Derivate constraint matrix

A constraint matrix $C$ is represented by the constraint conditions ($\Phi$) as

$$Cv = 0,$$

where $v$ is represented by the generalized velocity vector.

A constraint matrix $C_{w_{1h}}$ for the module is derived as following steps. The velocity $v_w$ and the position vector $x_w$ are defined as

$$v_w = \begin{bmatrix} \omega & v_r & \dot{x}_g \end{bmatrix}^T$$

$$= \begin{bmatrix} \omega_x & \omega_y & \omega_z & v_x & v_y & v_z & \dot{x}_g & \dot{y}_g & \dot{z}_g \end{bmatrix}^T$$
Figure 8: Illustration of constraints about $\Phi_7$

$$x_w = \left[ \Theta \ x \ x_y \right]^T$$

$$= \left[ \Theta \ y \ z \ x_y \ y_y \ z_y \right]^T.$$  

Additionally, the relation between $v_w$ and $x_w$ is represented by (2), (4), as follows

$$v_w = A_G \dot{x}_w$$

(17)

$$= \left[ \begin{array}{c} A_\theta \ Z_{3x3} \ Z_{3x3} \\ Z_{3x3} \ A^{-1} \ Z_{3x3} \\ Z_{3x3} \ Z_{3x3} \ I_{3x3} \end{array} \right] \begin{bmatrix} \dot{x} \\ \dot{x} \end{bmatrix}$$

where, $Z_{3x3}$ is matrix filled with zero, and the size is $3 \times 3$. $I_{3x3}$ is the identities matrix, and the size is $3 \times 3$.

The conditions of ($\Phi_1$, $\Phi_2$, $\Phi_3$, $\Phi_4$) are holonomic condition. Thus a constraint matrix $C_{wh}$ is represented by partial differential with respect to $x_w$;

$$\begin{bmatrix} \frac{\partial}{\partial x_w} \Phi_1 \\ \frac{\partial}{\partial x_w} \Phi_2 \\ \frac{\partial}{\partial x_w} \Phi_3 \\ \frac{\partial}{\partial x_w} \Phi_4 \end{bmatrix} x_w = 0 \Leftrightarrow \begin{bmatrix} \frac{\partial}{\partial x_w} \Phi_1 \\ \frac{\partial}{\partial x_w} \Phi_2 \\ \frac{\partial}{\partial x_w} \Phi_3 \\ \frac{\partial}{\partial x_w} \Phi_4 \end{bmatrix} A_G^{-1} v_w = 0$$

$$C_{wh} := \begin{bmatrix} \frac{\partial}{\partial x_w} \Phi_1 \\ \frac{\partial}{\partial x_w} \Phi_2 \\ \frac{\partial}{\partial x_w} \Phi_3 \\ \frac{\partial}{\partial x_w} \Phi_4 \end{bmatrix} A_G^{-1}.$$  

(18)

The condition of ($\Phi_5$) is non-holonomic condition. A constraint matrix $C_{wh}$ is represented by the constraint condition partial differentiated by $v_w$, as follows

$$\frac{\partial}{\partial v_w} \Phi_5 v_w = 0 \Leftrightarrow C_{wh} v_w = 0$$

$$C_{wh} := \frac{\partial}{\partial v_w} \Phi_5.$$  

(19)
A constraint matrix $C_w$ of the module is represented by (18) and (19);

$$
\begin{bmatrix}
  C_{wh} \\
  C_{enw}
\end{bmatrix} v_w = 0 \Leftrightarrow C_w v_w = 0,
$$

$$
C_w := \begin{bmatrix}
  C_{wh} \\
  C_{enw}
\end{bmatrix}.
$$

Additionally, decompose the generalized velocity as follows

$$
C_w \begin{bmatrix}
  v_{wx} \\
  v_{wy}
\end{bmatrix} = 0
$$

$$
\begin{bmatrix}
  C_{wx} & C_{wy}
\end{bmatrix} \begin{bmatrix}
  v_{wx} \\
  v_{wy}
\end{bmatrix} = 0
$$

$$
v_{wx} := \begin{bmatrix}
  \omega_x & \omega_y & \omega_z & v_x & v_y & v_z
\end{bmatrix}^T
$$

$$
v_{wy} := \begin{bmatrix}
  x_y & y_y & z_y
\end{bmatrix}^T.
$$

A constraint matrix $C_b$ for each modules is derived as following steps. The velocity $v_{b(i)}$ and the position vector $x_{b(i)}$ are defined as

$$
v_{b(i)} = \begin{bmatrix}
  \omega_{x(i)} & \omega_{y(i)} & \omega_{z(i)} & v_{x(i)} & v_{y(i)} & v_{z(i)}
\end{bmatrix}^T
$$

$$
= \begin{bmatrix}
  \omega_x & \omega_y & \omega_z & v_x & v_y & v_z \omega_x(i+1) \omega_y(i+1) \omega_z(i+1) v_x(i+1) v_y(i+1) v_z(i+1)
\end{bmatrix}^T
$$

$$
x_{b(i)} = \begin{bmatrix}
  \Theta(i) & x(i) & \Theta(i+1) & x(i+1)
\end{bmatrix}^T
$$

$$
= \begin{bmatrix}
  \theta(i) \psi(i) \phi(i) x(i) y(i) z(i) \theta(i+1) \psi(i+1) \phi(i+1) x(i+1) y(i+1) z(i+1)
\end{bmatrix}^T.
$$

Additionally, the relation between $v_{b(i)}$ and $x_{b(i)}$ is represented by (2), (4);

$$
v_{b(i)} = A_{Gb} x_{b(i)}
$$

$$
= \begin{bmatrix}
  A_{0(i)} & Z_{3x3} & Z_{3x3} & Z_{3x3} & Z_{3x3} & Z_{3x3} & Z_{3x3} & A_{-1(i+1)} & x(i+1)
\end{bmatrix}
$$

$$
A_{Gb} := \begin{bmatrix}
  A_{0(i)} & Z_{3x3} & Z_{3x3} & Z_{3x3} & Z_{3x3} & Z_{3x3} & Z_{3x3} & A_{-1(i+1)} & x(i+1)
\end{bmatrix}
$$

where, $i$ is number of the module ($i = 1, 2, \ldots, n - 1$).
The conditions of \((\Phi_0, \Phi_T)\) are holonomic condition. Thus a constraint matrix \(C_b\) is represented by partial differential with respect to \(x_{bi}\);

\[
\left[ \frac{\partial \Phi_0}{\partial x_{bi}}, \frac{\partial \Phi_T}{\partial x_{bi}} \right] x_{bi} = 0 \Rightarrow \left[ \frac{\partial \Phi_0}{\partial x_{bi}}, \frac{\partial \Phi_T}{\partial x_{bi}} \right] A_{Gi}^{-1} j_{bi} = 0
\]

\[
C_b := \left[ \frac{\partial \Phi_0}{\partial x_{bi}}, \frac{\partial \Phi_T}{\partial x_{bi}} \right] A_{Gi}^{-1}
\]

additionally, decompose the generalized velocity as follows

\[
C_b \begin{bmatrix} v_{wx(i)} \\ v_{wx(i+1)} \end{bmatrix} = 0
\]

\[
\begin{bmatrix} C_{b(i)} & C_{b(i+1)} \end{bmatrix} \begin{bmatrix} v_{wx(i)} \\ v_{wx(i+1)} \end{bmatrix} = 0.
\]

As a results, the constraint matrix of \(n\)-links snake-like robot is defined as

\[
\begin{bmatrix}
C_{wx1} & Z_{5x6} & \ldots & Z_{5x6} \\
Z_{5x6} & C_{wx2} & \ldots & Z_{5x6} \\
\vdots & \vdots & \ddots & \vdots \\
Z_{5x6} & Z_{5x6} & \ldots & C_{wxn} \\
C_{b1} & Z_{4x6} & \ldots & Z_{4x6} \\
Z_{4x6} & C_{b2} & \ldots & Z_{4x6} \\
\vdots & \vdots & \ddots & \vdots \\
Z_{4x6} & Z_{4x6} & \ldots & C_{b(n-1)} \\
Z_{4x6} & C_{b(n-1)} & \ldots & C_{bn} \\
\end{bmatrix}
\begin{bmatrix} v_{wx1} \\ v_{wx2} \\ \vdots \\ v_{wxn} \\ v_{w1} \\ v_{w2} \\ \vdots \\ v_{wn} \end{bmatrix} = 0
\]

\[
\Rightarrow Cb = 0.
\]

### 2.4. Derive constraint dynamical system

The snake-like robot model using the constraint matrix is derived. Motion equation of the snake-like robot without any constraints is represented by (6),(8) and (12);

\[
M_s \ddot{x}_s = h_s
\]

\[
M_s := \text{diag}(J_1, M_1, \ldots, J_n, M_n, M_{g1}, \ldots, M_{gn})
\]

\[
v_s := \begin{bmatrix} \omega_1^T \\ \omega_2^T \\ \vdots \\ \omega_n^T \end{bmatrix}
\]

\[
h_s := \begin{bmatrix} n_1^T h_1^T \\ n_2^T h_2^T \\ \vdots \\ n_n^T h_n^T \end{bmatrix}
\]

In projection method, the constraint dynamical model is represented by (21), \(C\) and \(\lambda\);

\[
M_s \dot{\phi}_s = h_s + C^T \lambda
\]

Where, an independent velocity of the constraint dynamical model is defined as \(\dot{q} := [\omega_{x1} \omega_{y1} \omega_{z1} v_{y1}]^T\). Additionally, \(v_s\) is decomposed into \(\dot{q}\) part and another part. However,
in this case needs exchange $v_{x1}$ and $v_{y1}$. Hence, exchanging 4th row and 5th row of $v_s, h_s$, and exchange 4th row and 5th row, 4th column and 5th column of $M_s, C$.

$$v_s = \begin{bmatrix} q^T & v_D^T \end{bmatrix}^T$$  \hspace{1cm} (23)

$C$ is decomposed into $\dot{q}$ part and $v_D$ part.

$$C \begin{bmatrix} \dot{q} \\
 v_D \end{bmatrix} = 0$$

$$\begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} \dot{q} \\
 v_D \end{bmatrix} = 0$$

$$C_1 \dot{q} + C_2 v_D = 0$$

$$\Leftrightarrow v_D = -C^{-1}_2 C_1 \dot{q}$$  \hspace{1cm} (24)

The relation between $v_s$ and $\dot{q}$ is represented by (23), (24).

$$v_s = \begin{bmatrix} I_{4\times4} \\
 -C^{-1}_2 C_1 \end{bmatrix} \dot{q}$$  \hspace{1cm} (25)

where, $D$ is the orthogonal complement matrix to $C$ satisfying $CD = 0$. Additionally, $\dot{D}$ is derived as

$$\dot{D} := \begin{bmatrix} Z_{4\times4} \\
 -C^{-1}_2 CD \end{bmatrix}$$  \hspace{1cm} (26)

Multiplying (22) by $D^T$ from the left-hand side and substituting (25) into (22), the constant term with $\lambda$ can be vanished, and the function is represented as

$$D^T M_s D \ddot{q} = D^T (h_s - M_s \dot{q})$$  \hspace{1cm} (27)

$$\Leftrightarrow M_q \ddot{q} = h_q$$

$$M_q := D^T M_s D$$

$$h_q := D^T (h_s - M_s \dot{q})$$

### 3. Control System Design

Control law is designed for the snake-like robot to move on the curved surface and to reach a target position from any initial condition. The control law makes the snake-like robot winding motion during locomotion. Also, the control law does not need any reference trajectory from an initial position to a target position, that is, it requires only the target position. The control law is designed by State-Dependent Riccati Equation control strategy in [10].

The control law is based on the model derived in the previous sections. However, the state vector is concerned with the pseudo velocity, and is not to the physical values related to
the position and the angle. If the control variables are related to the generalized coordinate, it is better for designers to design the controller because it is intuitive. From the reason, we first define a vector including the attitude of the head as controlled variable:

$$x_c = [\theta_1 \psi_1 \phi_1]^T.$$  \hspace{1cm} (28)

In general, the generalized force vector $h_q$ contains the input term, and gravity term and other nonlinear terms. On the other hand, we should separate the generalized force vector $h_q$ into the input term and the others for applying SDRE control strategy to the snake-like locomotion control. Hence, $h_q$ is separated to

$$h_q = Eu + Fv_s + G,$$  \hspace{1cm} (29)

where

$$E := \frac{\partial}{\partial u} h_q$$
$$F := \frac{\partial}{\partial v_s} (h_q - Eu)$$
$$G := h_q - Eu - Fv_s$$

$$u = [\tau_{x1} \tau_{z1} \cdots \tau_{x(n-1)} \tau_{z(n-1)}]^T.$$

$x_c$ is the controlled variable, and the equation of motion of the snake-like robot is given by (27). Therefore we should make the relation between $x_c$ and $q$ clear. The derivatives of $x_c$ can be represented by

$$\dot{x}_c = A_c \dot{q}$$  \hspace{1cm} (30)

where

$$A_c := \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & Z_{6(n-1)\times1} & Z_{3(n-1)\times1} \\ 0 & 1 & -\tan \theta_1 & 0 & 0 & 0 & 0 & Z_{6(n-1)\times1} & Z_{3(n-1)\times1} \\ 0 & 0 & \sec \theta_1 & 0 & 0 & 0 & 0 & Z_{6(n-1)\times1} & Z_{3(n-1)\times1} \end{bmatrix}.$$

$A_c$ is a coordinate transformation matrix from the pseudo-velocity coordinate space to the global coordinate space consisting of especially $\dot{\theta}_1, \dot{\psi}_1, \dot{\phi}_1$.

Using the coordinate transformation, we introduce an augmented state space model of the snake-like robot is represented by (27), (28) and (29) as follows;

$$\begin{bmatrix} \dot{x}_c \\ \dot{q} \end{bmatrix} = \begin{bmatrix} Z & A_c D \\ Z & M_q \end{bmatrix} \begin{bmatrix} x_c \\ \dot{q} \end{bmatrix} + \begin{bmatrix} Z \\ \dot{N} \end{bmatrix} u,$$  \hspace{1cm} (31)

where $\dot{M}$ and $\dot{H}$ are derived from (27). Each expression is represented as

$$\dot{M} := (D^T M_q D)^{-1} \left( D^T F D - D^T M_q D \right)$$
$$\dot{H} := (D^T M_q D)^{-1} D^T E.$$
respectively. Finally, we define the structure of weighting matrices. The SDRE control strategy is very similar to an ordinary optimal control for linear systems. As the optimal control for linear system, the SDRE strategy also requires the state-space form, but which is state-dependent, and the criterion whose structure is similar to quadratic. In SDRE case, the weight matrices in the criterion are allowed state-dependent. Therefore, we consider the criterion with state-dependent weight matrices for the locomotion control of snake-like robot. We suppose that the weight matrices consists of the basic weight matrices and the state-dependent matrices. We also suppose to make the basic weight matrices act for the stability of the entire system, and to make the state-space weight matrices act for tuning the twining behavior of the robot.

\[
Q(x) = Q_1 + \begin{bmatrix} Z & Z \\ Z & \alpha(x)^T Q_2 \alpha(x) \end{bmatrix} \\
R(x) = R_1 + \begin{bmatrix} Z & Z \\ Z & \beta(x)^T R_2 \beta(x) \end{bmatrix}
\]

where \(Q_1\) and \(R_1\) are the basic weight matrices, and \(Q_2\) and \(R_2\) are core weight matrices concerned with the state-dependent ones in the forms of quadratic \(\alpha(x)^T Q_2 \alpha(x)\). \(Z\) means zero matrix with appropriate dimension. \(\alpha(x(i))\) and \(\beta(x(i))\) are derived from (22) as

\[
C^T \lambda = C^T (CM_q^{-1} C^T)^{-1} C \left( D q - M_q^{-1} h_q \right) \\
= \alpha q + \beta u \\
\alpha := C^T (CM_q^{-1} C^T)^{-1} C \left( D - M_q^{-1} F \right) \\
\beta := -C^T (CM_q^{-1} C^T)^{-1} C M_q^{-1} E
\]

The weight matrix \(\alpha(x)^T Q_2 \alpha(x)\) is represented as

\[
\alpha(x)^T Q_2 \alpha(x) = \begin{bmatrix} \alpha_{ox}^T & \alpha_{oy}^T & \alpha_{oz}^T & \alpha_{oy}^T \\ Z & Q_{ox} & Z & Z \\ Z & Z & Q_{oz} & Z \\ Z & Z & Z & Q_{oy} \end{bmatrix} \begin{bmatrix} \alpha_{ox} \\ \alpha_{oy} \\ \alpha_{oz} \\ \alpha_{oy} \end{bmatrix}
\]

where \(\alpha_{ox}\) is related to a constraint force in \(\omega_x\) direction. The other \(\alpha\) are defined as well. \(Q_{ox}\) is a weight matrix for \(\alpha_{ox}\), and the other \(Q\) are defined as well. Tuning the ratio of the weight matrices \(Q_{ox}, Q_{oy}, Q_{oz}\) and \(Q_{oy}\) falls out varying the twining behavior because the constraint force related to the weight matrix is evaluated much more in the criterion. Also, the weight matrix \(\beta(x)^T R_2 \beta(x)\) is described as

\[
\beta(x)^T R_2 \beta(x) = \begin{bmatrix} \beta_{ox}^T & \beta_{oy}^T & \beta_{oz}^T & \beta_{oy}^T \\ Z & R_{ox} & Z & Z \\ Z & Z & R_{oz} & Z \\ Z & Z & Z & R_{oy} \end{bmatrix} \begin{bmatrix} \beta_{ox} \\ \beta_{oy} \\ \beta_{oz} \\ \beta_{oy} \end{bmatrix}
\]

Tuning \(R_2\) leads to the similar result of \(Q_2\). The effect of the state-dependent weight matrices in the criterion with \(Q_2\) and \(R_2\) are demonstrated by the following numerical simulations.
4. Simulation

Numerical simulations are performed to verify the effectiveness of the snake-like robot model and the controller based on the model. In the simulations, the number of body modules is set 5, i.e. $n = 5$. The shape function defining the smooth surface is given by $x_g^2 + y_g^2 + z_g^2 = 4$, i.e. a sphere with a relatively large radius. Two weight matrices $Q_1$ and $Q_2$ of the criterion in SDRE controller design are considered for checking how the weight matrices affect to the behavior of the snake-like robot. $Q_1$ and $Q_2$ are given as in Table 1. As mentioned in the previous section, $Q_2$ is related to the weights for adjusting the constraint force including side-slipping. In the case 1, the controller does not care about all of the constraint force. In the case 2, the controller does. The numerical simulations have been performed by using MaTX [14, 15]. The solver for ordinary differential equations is chosen as the Runge-Kutta 4th and 5th order method. The parameters of a module of the snake-like robot are presented in Table 2. Also, the simulation configuration is in Table 3.

As a result, in both cases, the snake-like robot moves toward the target point at the top left corner on the surface by the proposed method. From Figures 10–12 and Figures 14–16, we found out its behavior seems like winding as real snakes. Finally the 1st module of the robot moves from the initial position to the target potion even though the surface is not flat but a curved surface as demonstrated in Figure 9 and Figure 13.

From the comparison among Figures 9–13 and Figures 12–16, the head in case 2 has reached the target position faster than case 1, and winding motion is suppressed than case 1. Therefore the locomotion can be improved by considering constraint force during moving. From the results, we conclude that the proposed method allows the snake-like robot to move even on the curved surface, converges the robot head from the initial position to the target position without any reference trajectory, and can take the constraint force into account by the criterion as weight matrices in order to improve the locomotion behavior increasing the speed and suppressing the winding motion.

Table 1: Weight matrices in case 1 (no attention to the constraint force) and case 2 (attention to the constraint force). “diag()” means a diagonal matrix with the entries.

|       | $Q_1$                  | $Q_2$                  |
|-------|------------------------|------------------------|
| case 1| diag(20, 30, 10, 1, 1, 1, 5) | diag(0, 0, 0, 0, 0, 0) for $\omega x$ |
|       |                         | diag(0, 0, 0, 0, 0, 0) for $\omega y$ |
|       |                         | diag(0, 0, 0, 0, 0, 0) for $\omega z$ |
|       |                         | diag(0, 0, 0, 0, 0, 0) for $v_y$ |
| case 2| diag(20, 30, 10, 1, 1, 1, 5) | diag(20, 20, 20, 20, 20) for $\omega x$ |
|       |                         | diag(20, 20, 20, 20, 20) for $\omega y$ |
|       |                         | diag(20, 20, 20, 20, 20) for $\omega z$ |
|       |                         | diag(20, 20, 20, 20, 20) for $v_y$ |
Table 2: Physical parameters

| Notation | Value       |
|----------|-------------|
| $j_{xi}$ | $1/4 m_i r_i^2$ |
| $j_{yi}$ | $1/2 m_i r_i^2$ |
| $j_{zi}$ | $1/4 m_i r_i^2$ |
| $m_i$    | 1           |
| $r_i$    | 0.05        |
| $l_i$    | 0.05        |

Table 3: Simulation configuration

| Simulation environment | MaTX (matc version 5.3.43) |
|------------------------|-----------------------------|
| ODE solver             | Runge-Kutta 4th and 5th method |
| ODE step size          | 0.001s                      |
| Simulation duration time | 8s                          |

5. Conclusion

In this paper, we have presented the three-dimensional model of a snake-like robot on a smooth surface in arbitrary curvature. The model has been derived by the Projection method. The controller for the snake-like robot winding and moving on the surface is designed by SDRE. The effectiveness of the proposed method has been confirmed through numerical simulation.

The Projection method plays a powerful role for deriving a complex model like a snake-like robot, and the method can confine the movement of snake-like robot. The designed controller is possible to operate the snake-like robot without path planning, which is an advantage of the proposed method. The control input can be tuned by varying the weight matrices of the criterion of LQR. The entire performance is tuned by adjusting the weighting matrix. Finally it has been confirmed that the snake-like robot winds and moving on the surface by the proposed controller based on the three-dimensional model.

References

[1] S. Hirose, *Biologically Inspired Robots – Snake-Like Locomotors and Manipulators*, New York: Oxford Univ. Press (1993).

[2] S. Hirose and A. Morishima, Design and Control of a Mobile Robot with an Articulated Body, The International Journal of Robotics Research, Vol.9, No.2, pp.99-114 (1990).

[3] G. Hicks and K. Ito, A Method for Determination of Optimal Gaits With Application to a Snake-Like Serial-Link Structure, *IEEE Transactions on Automatic Control*, Vol.50, Issue.9, pp.1291-1306 (2005).
Figure 9: Trajectory of the head module’s CoG in case 1

Figure 10: $\theta$ profile: case 1

Figure 11: $\psi$ profile: case 1

Figure 12: $\phi$ profile: case 1

Figure 13: Trajectory of the head module’s CoG in case 2

Figure 14: $\theta$ profile: case 2

Figure 15: $\psi$ profile: case 2

Figure 16: $\phi$ profile: case 2

[4] P. Prautsch and T. Mita, Control and Analysis of the Gait of Snake Robots, in *Proceedings of the 1999 IEEE International Conference on Control Applications*, Kohala Coast-Island of Hawaii, Hawaii, pp 502-507 (1999).
[5] F. Matsuno and H. Sato, Trajectory Tracking Control of Snake Robots Based on Dynamic Model, in Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, pp3029-3034 (2005).

[6] M. Tanaka and F. Matsuno, Control of 3-dimensional Snake Robots by Using Redundancy, in Proceedings of the 2008 IEEE International Conference on Robotics and Automation, Pasadena, CA, pp1156-1161 (2008).

[7] A A. Transeth, R I. Leine, C. Glocker, K Y. Pettersen and P. Liljebäck, Snake Robot Obstacle-Aided Locomotion: Modeling, Simulations, and Experiments, IEEE Transactions on Robotics, Vol.24, No.1, pp.88-104 (2008).

[8] M. Tesch, K. Lipkin, I. Brown, R. L. Hatton, A. Peck, J. Rembisz, and H. Choset, “Parameterized and Scripted Gaits for Modular Snake Robots,” Advanced Robotics, vol. 23, pp. 1131-1158, June 2009

[9] C. Wright and et al., “Motion Estimation of Snake Robots in Straight Pipes”, Proc. of 2013 IEEE International Conference on Robotics and Automation (ICRA), pp.5168-5173, 2013

[10] K. Watanabe, M. Iwase, S. Hatakeyama and T. Maruyama, “Control Strategy for a Snake-like robot based on Constrain Force and Verification by Experiment”, Proc. of IROS 2008, pp.1616-1623, 2008

[11] Y. Suzuki, M. Iwase, T. Sadahiro and S. Hatakeyama, “A squeezing controll for snake-like robots to climb up trees”, Proc. of SICE Annual Conference 2010, pp.1862-1867, 2010

[12] K. Arczewsk and W. Blajer, A Unified Approach to the Modelling of Holonomic and Nonholonomic Mechanical Systems, Mathematical Modelling of Systems, Vol.2-3, No.668, pp.157-174 (1996).

[13] W. Blajer, A Geometrical Interpretation and Uniform Matrix Formulation of Multi-body Systems Dynamics, ZAMM Z Angew Math Mech, Vol.81, No.4, 2004, pp.247-259 (2001).

[14] M. Iwase, S. Kojima, T. Sadahiro and S. Hatakeyama, Setting up MaTX-based Laboratory Experimental Systems - A Pendulum System with Windows OS, IFAC Proceedings Volumes, Vol. 42, Issue 24, Pages 216–221 (2010)

[15] M. Koga, MaTX/RtMaTX: a freeware for integrated CACSD, Proceedings of the 1999 IEEE International Symposium on Computer Aided Control System Design, HI USA, DOI: 10.1109/CACSD.1999.808690 (1999)