SNAKE ROBOT GAIT DECOMPOSITION AND GAIT PARAMETER OPTIMIZATION

Bongsub Song, Insung Ju and Dongwon Yun
Department of robotics engineering
Daegu Gyeongbuk Institute of Science and Technology
Daegu
Republic of Korea
{1}doorebong@dgist.ac.kr
{2}mech@dgist.ac.kr

ABSTRACT
This paper proposes Gait Decomposition (G.D), a method of mathematically decomposing snake movements, and Gait Parameter Gradient (GPG), a method of optimizing decomposed gait parameters. G.D is a method that can express the snake gait mathematically and concisely from generating movement using the curve function to the motor control order when generating movement of snake robot. Through this method, the gait of the snake robot can be intuitively classified into a matrix, as well as flexibly adjusting the parameters of the curve function required for gait generation. This can solve the problem that parameter tuning, which is the reason why it is difficult for a snake robot to practical use, is difficult. Therefore, if this G.D is applied to snake robots, various gaits can be generated with a few of parameters, so snake robots can be used in many fields. We also implemented the GPG algorithm to optimize the gait curve function as well as define the gait of the snake robot through G.D.

Keywords Bio-inspired Learning · Learning & Optimization for Intelligent Robot · Model Learning for Control

1 INTRODUCTION

In order to utilize the redundant degree of freedom of snake robots, researches such as efficient modular structures (Rollinson and Choset, 2016; Yamada and Hirose, 2010) and attaching new sensors (Gong et al., 2015) are being conducted. In addition, researches are being conducted on how to locomote the snake robot through pre-defined repetitive movements using a serpentine curve or Central Pattern Generator (CPG) because of a problem that the complete dynamics of the snake robot system is difficult to know (Bing et al., 2017; Crespi and Ijspeert, 2008; Dehghani and Mahjoob, 2009; Enner et al., 2012; Gong et al., 2016, 2015; Manzoor et al., 2020, 2019; Rollinson and Choset, 2016, 2013; Takemori et al., 2018; Yamada and Hirose, 2010).

Among them, a study on a motion generation method using a serpentine curve was usually used (Dehghani and Mahjoob, 2009; Enner et al., 2012; Gong et al., 2016, 2015; Rollinson and Choset, 2016, 2013; Takemori et al., 2018; Yamada and Hirose, 2010). The serpentine curve for snake robot, pioneeringly, was presented as a method for locomotion in the robotic system developed by the Hirose research team (Hirose, 1993). Following this basis, a study (Ohno and Hirose, 2001) has also been conducted to divide the gait of limbless locomotion robots using the serpentine curve. However, since then, representative studies using the snake robot gait have been conducted by the Choset research team (Enner et al., 2012; Gong et al., 2016, 2015; Rollinson and Choset, 2016, 2013, 2013; Tesch et al., 2011). Among their studies, the study (Rollinson and Choset, 2016) has been conducted to enable snake robots to operate in a pipe network by tuning gait parameters. Also, in this study (Rollinson and Choset, 2013), researchers have studied on the compliant control of snake robots. In addition, the study (Gong et al., 2016) had been conducted to simplify gait by optimizing parameters.
Without utilizing serpentine curves, studies have been conducted to implement the movement of snake robots with Central Pattern Generator (CPG) inspired by the biological natural circuit research field. Among them, the study (Manzoor et al., 2019) was conducted to implement serpentine gait and sidewinding gait using CPG. The study (Manzoor et al., 2020) confirmed that a wheeled snake robot can also be locomoted using CPG.

All of the above series of studies have similarities in that the repetitive movements of snake robots are classified as gait and locomoted.

However, the gaits of the snake robot do not have a clear classification criterion like that of quadrupedal walking (Chong et al., 2018; Hildebrand, 1965; Hirose et al., 1986; Weingarten et al., 2003).

In this problem, we tried to distinguish the gait of a snake robot by using the pattern of movement, such as the method of classifying the gait of the quadrupedal robot (Chong et al., 2018). However, unlike a quadrupedal walking robot that classifies gaits through the contact time between feet and ground, a snake robot needs to classify gaits in a different way from a quadrupedal walking robot for the reasoning characteristic of a snake robot that contacts the whole body with the ground (Barazandeh et al., 2007; Date and Takita, 2005; Umetani and Hirose, 1974; Weingarten et al., 2003; Wu and Ma, 2010).

To distinguish these features, we would like to describe a new methodology named Gait Decomposition (G.D) to generate patterns using time slots and to classify the gait of the snake robot. Through G.D, the gait of the snake robot can be intuitively distinguished, and parameters that need to be tuned can be easily identified. Tuning these parameters would be helpful for finding the optimal gait parameters for the snake robot system.

We also introduce the Gait Parameter Gradient (GPG) algorithm in this paper, a method for tuning decomposed parameters. This algorithm will allow us to find the optimal parameters for newly defined gait.

After this introduction section, section 2 describes Gait Decomposition (G.D), a method of decomposing a gait of a snake robot. In this section, it is possible to identify optimizable parameters of the snake robot. Section 3 introduces the Gait Parameter Gradient (GPG) algorithm, an algorithm that optimizes the identified parameters. This algorithm is an algorithm that tunes parameters decomposed by G.D method. Through this algorithm, we can find the optimal parameters for the newly defined gait. In section 4, the optimal gait parameter is applied to the actual robot and compared with the simulation results. And in section 5, we would like to discuss whether our inference is valid based on the resulting two case of data. Finally, in the conclusion section, section V, we will conclude this paper and describe further research directions.

## 2 Gait Decomposition

Most of the preceding studies have locomoted their robots with the force generated by creating its links curves using curve functions and moving joints to form shapes similar to curves (Chong et al., 2018; Crespi and Ijspeert, 2008; Dehghani and Mahjoob, 2009; Gong et al., 2015; Rollinson and Choset, 2013; Wu and Ma, 2010). However, above gait in this way is difficult to express the order in which the motor is operated. When dealing with gait in other biological research fields, the gaits are classified using periodicity such as the swing phase and contact phase of a limb (Hildebrand, 1965). Therefore, the control order of a limb is also an important factor constituting the gait. However, if the gait is configured using only a curve function like serpentic function, it is impossible to intuitively configure the motor control order. This is because it is difficult to use independent variables that determine the control order. In this problem, we devised a method to construct the gait of a snake robot using a matrix representing the order of control along with an existing curve function. This method was named Gait Decomposition.

### 2.1 Motion Matrix

G.D is a gait decomposition method designed to distinguish the gait of a snake robot that locomotes by contacting the whole body with the ground, unlike a legged robot. In the case of a legged robot, the gaits are classified through the pattern obtained by plotting the period during feet and the ground contacts each other in a graph. However, we thought that it would be more intuitive for snake robots to plot and distinguish the time the joints move than to distinguish the gait by the period they touch the ground, and it could be a criterion for determining gait clearly. Figure 1 above represents three gaits of the snake robot.

The gaits of our snake robot could be classified into three gaits as shown in Figure 1. The vertical axis of the graph represents the index of the motors, and the horizontal axis represents the period of time when the motor is operated. Currently, the gait is configured to operate one motor at one gait phase slot, but the gait can be configured in which several motors can be controlled at one gait phase slot in the future. The motion matrices for each gait can be defined
In the case of Inchworm gait, only joints in the dorsal direction are controlled. Therefore, in the case of our system, only the odd motor moves. The reason is that the motors are configured to be orthogonal to each other as shown in Figure 2. This could be expressed as a motion matrix $M_{\text{inch}}$ as follows.

Figure 1: Defined three gaits and the gaits’ pattern

Figure 2: Motor configuration of implemented snake robot
The M matrix of the Inchworm gait consists of 7 gait phase slots. Dorsal joints are controlled in order in each gait phase slot.

Serpentine Gait Matrix

Unlike Inchworm gait, serpentine gait moves all joints. In addition, serpentine gait controls all motors in the order of the arrangement of its joints. This could be expressed as a motion matrix $M_{ser}$ as follows.

$$M_{ser} \in \mathbb{R}^{14 \times k}, k = 14$$

$$M_{ser} = \begin{bmatrix}
1 & 0 & 0 & \cdots & 0 & 0 \\
0 & 1 & 0 & \cdots & 0 & 0 \\
0 & 0 & 1 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & 1 & 0 \\
0 & 0 & 0 & \cdots & 0 & 1 \\
\end{bmatrix} = I$$

As shown in equation (2), the M matrix of serpentine gait consists of 14 gait phase slots, which is the number of motors. In addition, since all motors are controlled in order, $M_{ser}$ has the same shape as the unit matrix $I$.

Sidewinding Gait Matrix

Sidewinding gait has a section in which motor control is shuffled in the motor control order, unlike two gaits in which motor control order is arranged. This is expressed as a motion matrix $M_{side}$ as follows.

$$M_{side} \in \mathbb{R}^{14 \times k}, k = 14$$

$$M_{side} = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 & 0 \\
1 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 1 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & 1 & 0 \\
\end{bmatrix}$$

As shown in equation (3), the M matrix of the sidewinding gait consists of 14 gait phase slots, and there are parts where the lateral joint precedes the dorsal joint.

2.2 Curve Function

To propel the snake robot, the target angle must be given along with the M matrix. In general, snake robots use the serpenoid curve function to generate a target angle. Based on this prior research, we named the curve
function for controlling the snake robot as a P function. In addition, three P functions have been defined for each gait. Currently, the P function used by our robot is a function of time and has the maximum displacement of the joint, the phase of the triangular function, and the time as parameters. The defined P functions are as follows.

**Inchworm Gait P function**

Inchworm gait does not move the lateral joint as shown in equation (1). Therefore, the P function can be constructed with three parameters. The P function of Inchworm gait is shown as equation (4).

\[
P_{\text{inch}}(t, A, \phi) = \bar{\theta}, \bar{\theta} \in \mathbb{R}^{14}: \text{Goal position of joints, } i: \text{i-th motor from head}
\]

\[
\bar{\theta} = \begin{cases} 
  i = \text{odd}, \theta_i = A \sin \left(\frac{2\pi}{m} t + i \cdot \phi\right) \\
  i = \text{even}, \theta_i = 0
\end{cases}
\]  

(4)

Since the lateral joint is not an odd-numbered motor, so that the non-odd-numbered joint, lateral joint, is always zero, so it does not move.

**Serpentine Gait P function**

Unlike Inchworm gait, Serpentine gait consists of five parameters because all joints move, and the P function can be derived as equation (5).

\[
P_{\text{ser}}(t, A_{\text{dor}}, \phi_{\text{dor}}, A_{\text{lat}}, \phi_{\text{lat}}) = \bar{\theta}, \bar{\theta} \in \mathbb{R}^{14}
\]

\[
\bar{\theta} = \begin{cases} 
  i = \text{odd}, \theta_i = A_{\text{dor}} \cdot \sin \left(\frac{2\pi}{m} t + (i - 1) \cdot \phi_{\text{dor}}\right) \\
  i = \text{even}, \theta_i = A_{\text{lat}} \cdot \sin \left(\frac{2\pi}{m} t + (i + \frac{1}{2}) \cdot \phi_{\text{lat}}\right)
\end{cases}
\]  

(5)

**Sidewinding Gait P function**

Sidewinding gait also causes all joints to move, and the P function of sidewinding gait is as follows as equation (6).

\[
P_{\text{side}}(t, A_{\text{dor}}, \phi_{\text{dor}}, A_{\text{lat}}, \phi_{\text{lat}}) = \bar{\theta}, \bar{\theta} \in \mathbb{R}^{14}
\]

\[
\bar{\theta} = \begin{cases} 
  i = \text{odd}, \theta_i = A_{\text{dor}} \cdot \sin \left(\frac{2\pi}{m} t + \frac{1}{2} \cdot \phi_{\text{dor}}\right) \\
  i = \text{even}, \theta_i = A_{\text{lat}} \cdot \sin \left(\frac{2\pi}{m} t + \frac{1}{2} \cdot \phi_{\text{lat}}\right)
\end{cases}
\]  

(6)

### 2.3 Motor Control Algorithm

With Gait Decomposition, the gait of the snake robot could be divided into M matrix and P function. The target angle of the joint can be obtained at a specific time \( t \) through the M matrix and P function of three gaits configured in the G.D method. We calculated the target angle of the joint every 0.01 seconds (10\( \text{ms} \)) for smooth movement of the snake robot. If this time interval is long, the smoothness decreases, and if too short, the motor does not reach to the calculated position. This could be expressed by equation (7) as follows.

\[
M = \begin{bmatrix}
  m_1 & m_2 & \cdots & m_k
\end{bmatrix}, m_k \in \mathbb{R}^{14}
\]
\[
G_k = \bar{\theta} \cdot m_k^T, \quad G_k \in \mathbb{R}^{14 \times 14}
\]

(7)

The time step \( k \) increases every 0.01 seconds. When the time step is increased, the index of the column vector of the motion matrix \( \mathbb{M} (m_k \text{: gait phase slot}) \) and the value of the \( P \) function will be changed. This makes it possible to control different motors at different target angles for each time step. This is expressed as an algorithm as follows as Algorithm 1.

**Algorithm 1 Gait Generation Algorithm**

Define
\[
\mathbb{M} \coloneqq \text{Gait motion matrix}
\]
\[
P \coloneqq \text{Gait curve function}
\]
where
\[
\mathbb{M} \in \mathbb{R}^{m \times n}, \quad P \rightarrow \bar{\theta}, \quad \bar{\theta} \in \mathbb{R}^m
\]

\((m \text{: number of motors}, n \text{: column of motion matrix})\)

Variables:
\[
k \coloneqq 1, 2, 3, \ldots, \infty
\]
\[
\bar{g} \coloneqq \{g_1, g_2, \ldots, g_n\}, \quad \bar{g} \in \bar{\theta}
\]

1: while !Stop do
2: \( k \leftarrow k \mod n \)
3: \( \bar{g} \leftarrow \text{diag}(P(\bar{x}) \cdot m_k^T) \)
4: for \( i \in 1, 2, \ldots, m \) do
5: if \( g_i \neq 0 \) then
6: \( G_i \leftarrow g_i \)
7: else
8: Pass
9: end if
10: end for
11: \( k \leftarrow k + 1 \)
12: Send \((i, G_i)\) to motor driver
13: end while

According to the Algorithm above, the motors mounted on the snake robot are controlled and the robot is propelled. To optimally control the snake robot, the parameters of the \( P \) function and the configuration of the \( \mathbb{M} \) matrix could be optimized in the next section. To this end, we could also define an optimization method that could optimize the parameters of the \( P \) function according to the requirements of the system.

### 3 Gait Parameter Gradient

Snake robots are highly nonlinear systems because they are propelled with multi-contact with the ground. Therefore, it is difficult to obtain complete system dynamics (Hasanzadeh and Tootoonchi, 2010; Rollinson and Choset, 2013; Umetani and Hirose, 1974; Wu and Ma, 2010). In order to optimize the gait parameter of the snake robot, there were many cases of manual tuning parameters at the field where the snake robot operated (Rollinson and Choset, 2016). However, with the recent development of various robot dynamics simulators, tools for conducting dynamic simulations by expressing contact dynamics between robot body and ground in the software environment have been improved and widely used. In our case, we used Mujoco simulator for gait dynamics simulation. We implemented an algorithm that optimizes gait parameters using these dynamics simulators. also, we named this algorithm as Gait Parameter Gradient.

Prior to optimization, a utility function that evaluates the gait performance should be defined. We selected speed, straightness, and total displacements of joints as factors to evaluate gait’s performance. Among the above three factors, we mainly considered about straightness. This utility function can be freely reconstructed by gait designers in the future. In this paper, the following utility functions, \( U \), were used.

\[
U(k, P(k, \bar{\theta}, \mathbb{M}) = i \cdot \Delta x - j \cdot |\Delta y| - l \cdot |\frac{\Delta y}{\Delta x}| + m \cdot \int_{t_0}^{t_f} \|G_k\|_1 \, dk
\]

(8)

According to equation (8), the utility function evaluates the gait with system state after 1000 steps \((t_f = 10 \text{ sec})\). Since the utility function always returns the scalar value, the optimal gait parameter can be obtained by finding a parameter
### Table 1: Parameters in $\tilde{\theta}$

| Parameter notation | Description                             | Range     |
|--------------------|-----------------------------------------|-----------|
| $k$                | time step                               | $1, 2, \ldots, \infty$ |
| $A_{\text{dor}}$   | Maximum amplitude of dorsal joints      | $(0, 90)$ |
| $\phi_{\text{dor}}$ | Phase displacement between dorsal joints | $[0, 360)$ |
| $A_{\text{lat}}$   | Maximum amplitude of lateral joints     | $(0, 90)$ |
| $\phi_{\text{lat}}$ | Phase displacement between lateral joints | $[0, 360)$ |

that maximizes this value. In the utility function, $i$, $j$, $l$ and $m$ are performance indexes and can be modified according to the requirements of the system.

$$\max_{\tilde{\theta}} U(k, P(k, \tilde{\theta}), M)$$

subject to

$$A_i : (0, 90),$$

$$\phi i : [0, 360)$$

Here, $A$ and $\phi$ are subsets of $\tilde{\theta}$.

$$A, \phi \subset \tilde{\theta}$$

The optimization algorithm using the above conditions is as follows as algorithm 2.

**Algorithm 2 Gait Parameter Gradient Algorithm**

Define

- $U(x) :=$ Utility functional of the gait

**Variables:**
- $x :=$ Gait parameters, $\bar{x} \in \{A, \phi\}$, $\bar{x} \subset \mathbb{R}^n$
- $\hat{u} :=$ Direction of $\bar{x}$
- $\hat{e}_i := i$-th unit vector of $\bar{x}$
- $\lambda :=$ Step size

1: while $U(x^k) - U(x^{k-1}) > \varepsilon$, $(\varepsilon \neq 0)$ do
2:   for $i \in 1, 2, \ldots, n$ do
3:     $dx_i \leftarrow x + \alpha \cdot \hat{e}_i$
4:     if $U(x + dx_i) - U(x) > 0$ then
5:       $\hat{u}_i \leftarrow 1$
6:     else
7:       $\hat{u}_i \leftarrow -1$
8:   end if
9:   end for
10: $x^{k+1} \leftarrow x^k + \lambda \cdot \hat{u}$
11: end while
12: $x^* \leftarrow x^{k-1}$
13: Return $x^*$

We referred to the Nelder-Mead algorithm as an automatic differentiation method to find the gradient of the utility function. The Nelder-Mead algorithm is an algorithm that uses a simplex to find the gradient of a multivariate function.
without a model. Using this algorithm, it has been proven that optimal variables can be found in a heuristic way (Tesch et al., 2011; Weingarten et al., 2003; Yosinski et al., 2011).

The optimal parameters for two gaits (serpentine, sidewinding) could be obtained through the algorithm. The optimal parameters and gradients for the two gaits are shown in Figures 3 and 4 below.

By optimizing the serpentine gait with the above algorithm 2, we could find that the utility function was maximized when $A_{dor} = 39.8^\circ$, $\phi_{dor} = 189.9^\circ$, $A_{lat} = 66.5^\circ$ and $\phi_{lat} = 160.9^\circ$, and the value was $U(x^*) = 3565$. This is expressed as a red line in the trajectory graph of figure 3. Besides, it is expressed as a red dot in the surface graph.

Two control group parameters were selected for comparison with the optimal parameter obtained from the algorithm. These control parameters were obtained by changing the elements in dorsal parameter space ($A_{dor}$, $\phi_{dor}$) and lateral parameter space ($A_{lat}$, $\phi_{lat}$), respectively.

In the trajectory graph, the yellow line represents the gait trajectory of the control group 1. As can be seen from the yellow dot of the surface graph, the gait parameters of control group 1 were $A_{dor} = 40.7^\circ$, $\phi_{dor} = 191.1^\circ$, $A_{lat} = 66.5^\circ$ and $\phi_{lat} = 160.9^\circ$, and the utility function was $U(x_1) = 1232$. 

Figure 3: Trajectory of serpentine gait and utility function surface graph of serpentine gait
In addition, the green line represents the gait trajectory of the control group 2. As can be seen from the green dot of the surface graph, the gait parameters of control group 2 were $A_{dor} = 39.8^\circ$, $\phi_{dor} = 189.9^\circ$, $A_{lat} = 67.1^\circ$ and $\phi_{lat} = 160.3^\circ$, and the utility function was $U(x_{c2}) = 269$.

Also as shown in the figure above, lateral phase is the most dominant parameter of serpentine gait. Utility value is rapidly decreased according to this parameter. On the other hands, most subservient parameter is dorsal amplitude.

Likewise, sidewarding gait could be optimized. We found that the utility function was maximized when $A_{dor} = 52.76^\circ$, $\phi_{dor} = 319.65^\circ$, $A_{lat} = 72.07^\circ$ and $\phi_{lat} = 262.95^\circ$ and the value was $U(x^*) = 4268$. As well as serpentine gait, the dominant parameter of sidewarding gait is lateral phase ($\phi_{dor}$).

In addition to optimal parameter, we could select two control group parameters as well. Those two parameters are selected by changing the elements in dorsal parameter space ($A_{dor}, \phi_{dor}$) and lateral parameter space ($A_{lat}, \phi_{lat}$), respectively.

In the trajectory graph of Figure 4, the yellow line represents the gait trajectory of the control group 1. As can be seen from the yellow dot of the surface graph, the gait parameters of control group 1 were $A_{dor} = 52.16^\circ$, $\phi_{dor} = 318.15^\circ$, $A_{lat} = 72.07^\circ$ and $\phi_{lat} = 262.95^\circ$, and the utility function was $U(x_{c1}) = 3160$.

In addition, the green line represents the gait trajectory of the control group 2. As can be seen from the green dot of the surface graph, the gait parameters of control group 2 were $A_{dor} = 52.76^\circ$, $\phi_{dor} = 319.65^\circ$, $A_{lat} = 72.67^\circ$ and $\phi_{lat} = 261.75^\circ$, and the utility was $U(x_{c2}) = -746$. 
As shown in Figures 3 and 4 above, the gradient of the utility function value for gaits is expressed using a simulator. These gradient graphs are derived from equation (8).

4 Experiment

As described in the previous section, we were able to obtain optimal parameters for 2 gaits through optimization algorithms.

In this section, we are going to check whether the newly defined gait by our proposed Gait Decomposition (G.D) and Gait Parameter Gradient (GPG) method can be applied and utilized in actual robots by comparing the analysis results with the experimental results. If the optimal parameters obtained through simulation get actually maximum value of utility function, and if the gait trajectory is similar to that of the simulation, it can be said that our methodology is worth applying to actual snake robots.

To this end, we experimented the gait by changing gait parameters as in the figures 3 and 4 and obtained simulation results.
Figure 5 is the image obtained by driving a actual snake robot by applying the optimal parameters of each gait obtained in the previous section. We were able to film our snake robot by fixing camera to ceiling. The frame rate of the video was 60 fps. Therefore, it was possible to obtain about 600 position data samples through a 10-second video. However, in the case of simulation, since there are 1,000 data samples, it is necessary to compare the two data with the same number of data samples through an additional up-sampling methods.

First, Figure 6 shows the experimental result for the serpentine gait. This graph shows the trajectory of serpentine gait for 10 seconds. Three experimental results were obtained by applying the two control parameters and the optimal parameters from the previous section shown as Figure 3 above.

In Figure 6, the optimal parameters and control parameters are the same as those in Figure 3. As with the optimized result in previous section, the maximum utility value was obtained from the optimal parameter. In addition, the shape of the gait trajectory obtained when applying the control parameters 1 and 2 was similar to the result obtained in the simulation.

This means that the gait parameters of the snake robot can be effectively optimized with G.D and GPG.
Next, Figure 7 shows the experimental results for the sidewinding gait. Similarly to Figure 6, this experiment also showed the trajectory of sidewinding gait. Through the experiment, it was confirmed that the maximum utility value is obtained by optimized parameter on figure 4 for sidewinding gait.

5 Discussion

From Figures 6 and 7 of the previous experimental section, the trajectory of the snake robot could be confirmed. However, since this result is not perfectly identical to the trajectory we obtained through simulation, we would like to discuss the simulation and experimental results more precisely in this section.

Comparing the simulation results in Figures 3 and 4 and the experimental results in Figure 6 and 7, the two gait trajectories do not perfectly match even though the trajectories have been seen similar. We assumed that this difference
is due to the environment variables of the simulator. To compare the two results by minimizing the simulation error, a scale factor was multiplied by the experimental data. The experimental result data multiplied by the scale factor are shown in Figure 8 and 9.

The scale factor $F_s$ is the ratio of the displacement of the final state between the analysis and experiment through the data obtained by each gait parameter. This can be expressed as follows.

$$F_s = \sqrt{\frac{P_x(t_f)^2 + P_y(t_f)^2}{Q_x(t_f)^2 + Q_y(t_f)^2}}, \quad P := \text{simulation data}, \quad Q := \text{experiment data}$$

$$Q'(t) = F_s \cdot Q(t), \quad Q' := \text{scaled experiment data}$$

Since the scaled experimental data is multiplied by a scalar value, the trend of the data could be maintained.

First, from Figure 8, which shows three serpentine gait parameter trajectories, intuitively, the three types of parameters have been seen similar to each group. Also, in Figure 9, which shows the trajectory of sidewinding, the results of all kinds of parameters seem to have some similarities.

As shown in Figures 8 and 9, the experimental data and the simulation data do not perfectly match even if we multiplied scale factor to the experiment data. This kind of problem could not be solved by very fine tuning of the simulator’s environment variable. This is because, when measuring data, noise is also measured, and accurate environment variables cannot be applied to the simulator.

Therefore, in order to find the meaning of the optimization result, several perspectives are needed to judge additional similarity beyond the scale factor. In order to show that our optimization results are consistently useful even when applied to actual experiments, we would like to show similarities in the following three perspectives. The first is the norm similarity, the second is the direction similarity, and the last is the utility value similarity.

- Norm similarity
  A vector is needed to find the L2 Norm. For similarity comparison, we created a $\hat{y}$ to store the heading angle. The $\hat{y}$ can be calculated like the following equation.
Running Title for Header

Figure 9: Scaled sidewinding gait trajectories graph by gait parameters

\[ \hat{y} = \{ \theta_i \mid \theta_i = \arccos \left( \frac{x_i}{\sqrt{x_i^2 + y_i^2}} \right) \}, \text{ (for } i = 1, 2, \ldots, n) \]

Through \( \hat{y} \), it is possible to obtain \( N_{\text{gait}} \), which is the L2 norm of the error vector with respect to the goal direction as below.

\[ N_{\text{gait}} = ||y - \hat{y}||_2, \ y := \text{Goal direction} \]

The L2 Norm for each gait parameter obtained from experiments and simulations could be calculated as shown in Table 2. The difference in \( N_{\text{gait}} \) values between the simulation results and the experimental results in the optimal parameter is not large. In the case of the serpentine gait, there was a difference of about 3.7% in the optimal parameter, and in the case of the sidewinding gait, there was a difference of 3.8%. Through this perspective, we can confirm that the movement of the gait to which the optimal parameter is applied is moving in the direction we intended.

- **Direction (Heading) similarity**
  The \( N_{\text{gait}} \) value indicates the direction error, whereas the direction indicates which direction the system moved in the final state. We made the snake robot move using gait for 10 seconds (\( t_f \approx 10 \text{sec.} \)) in both experiments and simulations. Direction can be obtained through displacement for 10 seconds. Direction can be obtained by the following equation.

\[ \Phi = \arccos \left( \frac{x_{tf}}{\sqrt{x_{tf}^2 + y_{tf}^2}} \right), \ (x_{tf}, y_{tf} := \text{Each axis displacement at } t_f) \]

As can be seen in Table 2 below, the parameters of all groups, except the serpentine control group 2, showed a difference within 10 degrees. When the serpentine control group 2 parameter is applied to serpentine gait, there was a big difference in direction because it moves unstable in both experiments and simulations.

- **Utility value similarity**
  There are absolute difference in values of utility function on all 6 samples of parameters. However, the differences in relative utility values between all three groups are similar. For example, in the case of the serpentine
gait, utility values are shown in the order of $U(x^*) > U(x_{c2}) > U(x_{c1})$, ($x^*$ := optimized parameter, $x_{c1}$ := control group 1, $x_{c2}$ := control group 2) in both simulation and experimental results. Similarly, in the case of the sidewinding gait, the utility values appear in the order of $U(x^*) > U(x_{c1}) > U(x_{c2})$. This shows that the absolute utility values are different, but the relative utility values are similar to the simulation results and experimental results.

As above, it was possible to check the similarity between the simulation results and the experimental results from the three perspectives. Through the similarity of simulation results and experimental results, it is expected that it will be possible to optimize other gaits beyond optimizing the three gaits introduced in this paper.
### Table 2: Simulation and experiment result data

| Time step size (sec.) | 0.01 | 1000 | End step | 10 | 10.3567 | 10 | 10.4733 | 10 | 10.3067 | 10 | 10.5400 | 10 | 10.4733 | 10 | 10.7383 |
|-----------------------|------|------|----------|----|---------|----|---------|----|---------|----|---------|----|---------|----|---------|
| Total time elapsed (sec.) | 7719.1577 | 2761.5710 | -745.7709 | 72.67 | 3.8634 | 4268.3715 | 1.1625 | 0 | 2.4509 | 0.0001 | -0.4303 | 4268.3715 | 1.1625 |
| φA (deg.) | 39.8 | 39.8 | 40.7 | 39.8 | 52.76 | 52.76 | 52.16 | 52.16 | 52.76 | 52.76 | 52.76 | 52.76 | 52.76 |
| φlat (deg.) | 189.9 | 189.9 | 191.1 | 189.9 | 319.65 | 319.65 | 318.15 | 318.15 | 319.65 | 319.65 | 319.65 | 319.65 | 319.65 |
| Δr (m) | 6.65 | 66.5 | 66.5 | 67.1 | 72.07 | 72.07 | 72.07 | 72.07 | 72.07 | 72.07 | 72.07 | 72.07 | 72.07 |
| φα (deg.) | 160.9 | 160.9 | 160.9 | 160.9 | 262.95 | 262.95 | 262.95 | 262.95 | 262.95 | 262.95 | 262.95 | 262.95 | 262.95 |
| Δy (m) | 2.4879 | 0.7284 | 2.2476 | 0.8941 | 0.2123 | -0.1791 | -0.2206 | 0.259 | 0.9474 | 1.323 | -0.5812 | -0.3935 | 0.01 |
| Δy (m) | -0.0359 | -0.1279 | -1.7757 | -0.7438 | -0.0083 | -0.0365 | 3.2667 | 4.0499 | 2.9762 | 2.8274 | 0.5536 | 0.326 | 0.01 |
| X mean (m) | 1.1040 | 0.3604 | 1.097 | 0.4697 | 0.0644 | -0.0549 | -0.0945 | 0.041 | 0.2375 | 0.3183 | -0.2039 | -0.1971 | 0.01 |
| Y mean (m) | -0.0258 | 0.0288 | -0.8485 | -0.3448 | 0.0088 | 0.0003 | 1.4972 | 1.8456 | 1.3855 | 1.4005 | 0.4113 | 0.2128 | 0.01 |
| Distance (m) | 2.4882 | 0.7395 | 2.8644 | 1.163 | 0.2125 | 0.1828 | 3.2741 | 4.0581 | 3.1234 | 3.1216 | 0.8027 | 0.511 | 0.01 |
| Direction (deg.) | 0.8897 | 9.9621 | 38.3101 | 39.7556 | 2.486 | 168.4749 | 93.8634 | 86.3413 | 72.3427 | 64.9243 | 136.3904 | 140.3633 | 0.01 |

| Gait parameters | Optimal-Sim. | Optimal-Exp. | CG1-Sim. | CG2-Sim. | Optimal-Sim. | Optimal-Exp. | CG1-Sim. | CG2-Sim. |
|-----------------|-------------|-------------|----------|----------|-------------|-------------|----------|----------|
| Time (sec.) | 0.01 | 1000 | 10 | 10.3567 | 10 | 10.4733 | 10 | 10.3067 |
| X-Y plane trajectory | 7719.1577 | 2761.5710 | -745.7709 | 72.67 | 3.8634 | 4268.3715 | 1.1625 | 0 | 2.4509 | 0.0001 | -0.4303 | 4268.3715 | 1.1625 |
| Utility value | 1500 | 1500 | 1500 | 1500 | 1500 | 1500 | 1500 | 1500 | 1500 | 1500 | 1500 | 1500 | 1500 |
| Goal direction (deg.) | 0 | 0 | 0 | 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 90 | 90 |
| Direction error (deg.) | -0.8897 | -9.9621 | -38.3101 | -39.7556 | -2.486 | -168.4749 | -3.8634 | 3.6587 | 17.6573 | 25.0757 | -46.3904 | -50.3633 | 0.01 |
| ||y−y'||2 | 1053.0660 | 1015.9238 | 1479.3427 | 1221.4047 | 2537.9419 | 3204.3877 | 486.6320 | 468.8549 | 802.1690 | 1235.8148 | 537.2109 | 529.5913 | 0.01 |
| (y−y') std | 29.1226 | 31.5253 | 20.1524 | 25.7645 | 38.4562 | 39.4045 | 13.3278 | 13.6917 | 19.6962 | 27.8389 | 16.9547 | 15.9536 | 0.01 |
| Scared data analysis | Fy | 3.3647 | 2.4629 | 1.1625 | 0.8068 | 1.0006 | 1.5708 | 0.01 |
| Ax | 2.4509 | 2.2021 | -0.2082 | 0.2090 | 1.3238 | -0.6181 | 0.01 |
| Ay | -9.9621 | -39.7556 | -168.4749 | 86.3413 | -64.9243 | 140.3633 | 0.01 |
| Direction* | 1015.9238 | 1221.4047 | 3204.3877 | 468.8549 | 1235.8148 | 529.5913 | 0.01 |
| ||y−y'||2 | -3166.25526 | 1081.192 | 87.21783 | 467.499075 | 3655.2463 | -485.3784 | 0.01 |
6 Conclusion

Snake robots is a redundant system that has many degrees of freedom. Therefore, it is very difficult to obtain acceleration or velocity of snake robot directly from the system dynamics. Therefore, many prior studies have used a method of moving joints repeatedly by pre-defined repetitive movements called gait or CPG.

However, so far, no research has been conducted on how to classify the gait of snake robots. In this paper, we presented a method of expressing gait with a P function and an M matrix to classify the gait of snake robots. The P function is a curve function that produces a goal angle of a joint over time, and the M matrix is a selection matrix that selects a motor to be driven at a specific time slot.

The M matrix, which is the selection matrix, helps to intuitively distinguish gaits. This paper shows three M matrices for three gaits, but more gaits could be defined as defining another M matrix. The curve function P is a multivariate function with several gait parameters. By adjusting the parameters of this P function, we can configure a gait with good performance.

In this paper, to show this P function parameter optimization process, we presented a Gait Parameter Gradient algorithm that defines a simple utility function with four performance indices and optimizes the parameters of the P function through the corresponding utility function value. Through this algorithm, optimal P function parameters for two gaits for our system could be obtained.

Through experiments, we tried to confirm the similarity of simulation results and experimental results. In order to determine the similarity, we were able to compare these two data by three perspectives. First, L2 norm similarity, second, the direction similarity, and finally the utility value similarity.

The $N_{gaits}$, which is the L2 norm of error vector between the goal direction and current direction, was a difference within 4% at optimized parameter. In addition, the direction difference between initial state and final state vector was within 10 degrees at all parameters of each gait except control group 2 parameters of serpentine gait. Finally, the relative order of values of utility function were same in both simulation and experimental results.

Through the three perspectives of the similarity, we were able to confirm that we could pre-optimize the gait using G.D and GPG methods with multi-body dynamics simulation software like Mujoco. Through this, we were able to optimize serpentine gait and sidewinding gait of our actual snake robot. At this moment, the optimized gait parameters were $A_{dor} = 39.8^\circ$, $\phi_{dor} = 189.9^\circ$, $A_{lat} = 66.5^\circ$ and $\phi_{lat} = 160.9^\circ$ at serpentine gait and $A_{dor} = 52.76^\circ$, $\phi_{dor} = 319.65^\circ$, $A_{lat} = 72.07^\circ$ and $\phi_{lat} = 262.95^\circ$ at sidewinding gait.

Through this, we confirmed that G.D and GPG are helpful even when applied to actual robots. Therefore, it is expected that a better performance gait will be obtained if the utility function that can express system performance well is configured in the future. Additionally, through this study, it is expected that snake robot researchers can easily distinguish the gaits and tune the gait easily.

In this paper, it does not include the process of optimizing motion matrix M but only optimizing the P function. In the future, it is expected that research to obtain the optimal M matrix can be conducted beyond optimizing the parameters of the P function by expanding the policy concept of reinforcement learning.

ACKNOWLEDGMENT

This material is based upon work supported by the Ministry of Trade, Industry & Energy(MOTIE, Korea) under Industrial Technology Innovation Program. No. 20003739.

References

[1] Barazandeh, F., Bahr, B., Moradi, A., 2007. How self-locking reduces actuators torque in climbing snake robots, in: 2007 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. Presented at the 2007 IEEE/ASME international conference on advanced intelligent mechatronics, pp. 1–6. https://doi.org/10.1109/AIM.2007.4412524

[2] Bing, Z., Cheng, L., Chen, G., Röhrbein, F., Huang, K., Knoll, A., 2017. Towards autonomous locomotion: CPG-based control of smooth 3D slithering gait transition of a snake-like robot. Bioinspir. Biomim. 12, 035001. https://doi.org/10.1088/1748-3190/aa644c
[3] Chong, B., Ozkan Aydin, Y., Gong, C., Sartoretti, G., Wu, Y., Rieser, J., Xing, H., Rankin, J., Michel, K., Nicieza, A., Hutchinson, J., Goldman, D., Choset, H., 2018. Coordination of back bending and leg movements for quadrupedal locomotion. https://doi.org/10.15607/RSS.2018.XIV.020

[4] Crespi, A., Ijspeert, A.J., 2008. Online Optimization of Swimming and Crawling in an Amphibious Snake Robot. IEEE Trans. Robot. 24, 75–87. https://doi.org/10.1109/TRO.2008.915426

[5] Date, H., Takita, Y., 2005. Control of 3D Snake-Like Locomotive Mechanism Based on Continuum Modeling, in: Volume 6: 5th International Conference on Multibody Systems, Nonlinear Dynamics, and Control, Parts A, B, and C. Presented at the ASME 2005 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, ASMEDC, Long Beach, California, USA, pp. 1351–1359. https://doi.org/10.1115/DETC2005-85130

[6] Dehghani, M., Mahjoob, M.J., 2009. A modified serpenoid equation for snake robots, in: 2008 IEEE International Conference on Robotics and Biomimetics. Presented at the 2008 IEEE International Conference on Robotics and Biomimetics, pp. 1647–1652. https://doi.org/10.1109/ROBIO.2009.4913248

[7] Enner, F., Rollinson, D., Choset, H., 2012. Simplified motion modeling for snake robots, in: 2012 IEEE International Conference on Robotics and Automation. Presented at the 2012 IEEE International Conference on Robotics and Automation, pp. 4216–4221. https://doi.org/10.1109/ICRA.2012.6225163

[8] Gong, C., I. Goldman, D., Choset, H., 2016. Simplifying Gait Design via Shape Basis Optimization, in: Robotics: Science and Systems XII. Presented at the Robotics: Science and Systems 2016, Robotics: Science and Systems Foundation. https://doi.org/10.15607/RSS.2016.XII.006

[9] Gong, C., Travers, M., Astley, H.C., Goldman, D.I., Choset, H., 2015. Limbless locomotors that turn in place, in: 2015 IEEE International Conference on Robotics and Automation (ICRA). Presented at the 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 3747–3754. https://doi.org/10.1109/ICRA.2015.7139720

[10] Hasanzadeh, S., Tootoonchi, A.A., 2010. Ground adaptive and optimized locomotion of snake robot moving with a novel gait. Auton. Robots 28, 457–470. https://doi.org/10.1007/s10514-010-9179-y

[11] Hildebrand, M., 1965. Symmetrical Gaits of Horses. Science 150, 701–708.

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

[13] Hirose, S., Kikuchi, H., Umetani, Y., 1986. The standard circular gait of a quadruped walking vehicle. Adv. Robot. 1, 143–164. https://doi.org/10.1163/156855386X00058

[14] Manzoor, S., Cho, Y.G., Choi, Y., 2019. Neural Oscillator Based CPG for Various Rhythmic Motions of Modular Snake Robot with Active Joints. J. Intell. Robot. Syst. 94, 641–654. https://doi.org/10.1007/s10846-018-0864-y

[15] Manzoor, S., Khan, U., Ullah, I., 2020. Serpentine and Rectilinear Motion Generation in Snake Robot Using Central Pattern Generator with Gait Transition. Iran. J. Sci. Technol. Trans. Electr. Eng. 44, 1093–1103. https://doi.org/10.1007/s40998-019-00301-8

[16] Ohno, H., Hirose, S., 2001. Design of slim slime robot and its gait of locomotion, in: Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180). Presented at the Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180), pp. 707–715 vol.2. https://doi.org/10.1109/IROS.2001.976252

[17] Rollinson, D., Choset, H., 2016. Pipe Network Locomotion with a Snake Robot. J. Field Robot. 33, 322–336. https://doi.org/10.1002/rob.21549

[18] Rollinson, D., Choset, H., 2013. Gait-based compliant control for snake robots, in: 2013 IEEE International Conference on Robotics and Automation. Presented at the 2013 IEEE International Conference on Robotics and Automation, pp. 5136–5143. https://doi.org/10.1109/ICRA.2013.6631311

[19] Takemori, T., Tanaka, M., Matsuno, F., 2018. Gait Design for a Snake Robot by Connecting Curve Segments and Experimental Demonstration. IEEE Trans. Robot. 34, 1384–1391. https://doi.org/10.1109/TRO.2018.2830346

[20] Tesch, M., Schneider, J., Choset, H., 2011. Using response surfaces and expected improvement to optimize snake robot gait parameters, in: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. Presented at the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1069–1074. https://doi.org/10.1109/IROS.2011.6095076

[21] Umetani, Y., Hirose, S., 1974. Biomechanical Study of Serpentine Locomotion, in: Biomechanical Study of Serpentine Locomotion | SpringerLink, International Centre for Mechanical Sciences. Springer, pp. 171–184.
[22] Weingarten, J.D., Buehler, M., Groff, R.E., Koditschek, D., 2003. Gait Generation and Optimization for Legged Robots 9.

[23] Wu, X., Ma, S., 2010. CPG-based control of serpentine locomotion of a snake-like robot. Mechatronics 20, 326–334. https://doi.org/10.1016/j.mechatronics.2010.01.006

[24] Yamada, H., Hirose, S., 2010. Approximations to continuous curves of Active Cord Mechanism made of arc-shaped joints or double joints, in: 2010 IEEE International Conference on Robotics and Automation. Presented at the 2010 IEEE International Conference on Robotics and Automation, pp. 703–708. https://doi.org/10.1109/ROBOT.2010.5509938

[25] Yosinski, J., Clune, J., Hidalgo, D., Nguyen, S., Zagal, J.C., Lipson, H., 2011. Evolving Robot Gaits in Hardware: the HyperNEAT Generative Encoding Vs. Parameter Optimization 9.