



A Gait Pattern Generator for Closed-Loop Position Control of a Soft Walking Robot

Lars Schiller^{1*}, Arthur Seibel² and Josef Schlattmann¹

¹ Workgroup on System Technologies and Engineering Design Methodology, Hamburg University of Technology, Hamburg, Germany, ² Fraunhofer Research Institution for Additive Manufacturing Technologies IAPT, Hamburg, Germany

OPEN ACCESS

Edited by:

Concepción A. Monje,
Universidad Carlos III de
Madrid, Spain

Reviewed by:

Cosimo Della Santina,
Massachusetts Institute of
Technology, United States
Chaoyang Song,
Southern University of Science and
Technology, China

*Correspondence:

Lars Schiller
lars.schiller@tuhh.de

Specialty section:

This article was submitted to
Soft Robotics,
a section of the journal
Frontiers in Robotics and AI

Received: 18 February 2020

Accepted: 02 June 2020

Published: 02 July 2020

Citation:

Schiller L, Seibel A and Schlattmann J
(2020) A Gait Pattern Generator for
Closed-Loop Position Control of a Soft
Walking Robot. *Front. Robot. AI* 7:87.
doi: 10.3389/frobt.2020.00087

This paper presents an approach to control the position of a gecko-inspired soft robot in Cartesian space. By formulating constraints under the assumption of constant curvature, the joint space of the robot is reduced in its dimension from nine to two. The remaining two generalized coordinates describe respectively the walking speed and the rotational speed of the robot and define the so-called velocity space. By means of simulations and experimental validation, the direct kinematics of the entire velocity space (mapping in Cartesian task space) is approximated by a bivariate polynomial. Based on this, an optimization problem is formulated that recursively generates the optimal references to reach a given target position in task space. Finally, we show in simulation and experiment that the robot can master arbitrary obstacle courses by making use of this gait pattern generator.

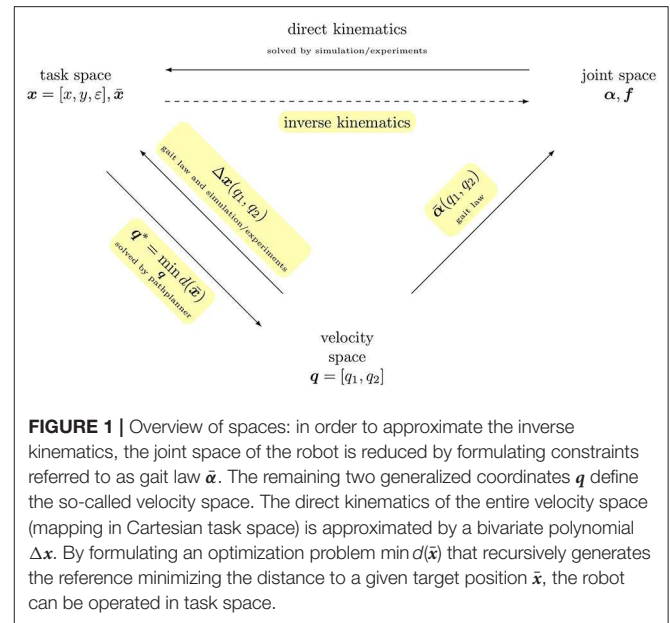
Keywords: mobile robotics, gait pattern generator, closed-loop position control, gecko-inspired soft robot, locomotion controller

1. INTRODUCTION

Soft robotics is an emerging field in the robotics sciences and enjoys increasing attention in the scientific community (Bao et al., 2018). An important part of this field is mobile soft robotics, which allows locomotion in unknown and unstructured (Katzschmann et al., 2018) as well as potentially dangerous environments (Tolley et al., 2014). In order to navigate a robot through any environment, some sort of feedback is needed. As discussed in Santina et al. (2017), high gain feedback control results in good tracking performance, but imposes a reduction in the compliance of the controlled system. Therefore, it takes away the essential characteristic and greatest advantage of a soft robot—its softness (Rus and Tolley, 2015). When it comes to soft robots, usually the dynamics of inputs are indirectly coupled with the dynamics of outputs and the coupling is time-delayed (PneuNets: pressure—angle, SMA: heat—contraction, refer to Lee et al., 2013). In order to take this into account, a cascaded control architecture has been established (see, e.g., Marchese et al., 2014; Hofer and D'Andrea, 2018). In the case of pneumatically operated robots, the inner loop controls the pressure and the outer loop controls the pressure reference (see also Figure 11B). In order to preserve softness, the feedback gain of the outer control loop needs to be low. Most of the pressure reference should therefore be generated by a feed forward term (Santina et al., 2017). There is a trend to implement the feed forward term by using Iterative Learning Control (Bristow et al., 2006; see, e.g., Santina et al., 2017; Zhang and Polygerinos, 2018; Hofer et al., 2019). As shown in Santina et al. (2020), the typical soft properties of a soft robot can also be preserved with a model-based feed forward term when doing position control.

All the soft robots discussed so far are stationary. Thus, position control refers to the position of the end effector and not to the position of the entire robot. However, the same principles are also valid for mobile soft robots. Most mobile soft robots, such as in Shepherd et al. (2011), Godage et al. (2012), Tolley et al. (2014), Qin et al. (2019), and Schiller et al. (2019), are feed forward-controlled with predefined gait patterns. In order to enable such robots to move autonomously even in unknown terrain, a locomotion controller is needed that can generate any gait path. For solving this task, different methods have been employed, such as sine generators, central pattern generators (CPG), predefined trajectories, finite state machines, or heuristic control laws (Pratt et al., 2001). An example for sine generator-based locomotion control is presented in Horvat et al. (2015, 2017) for a salamander-like robot. The method enables to operate the robot by only two drive signals, i.e., forward and rotational speed. The main contribution here is the skilful synchronization of spine and legs motion, which is very robot-specific. In Ijspeert (2008), the suitability of central pattern generators, i.e., biologically inspired neural circuits capable of producing coordinated patterns for robot's locomotion, are discussed. It is concluded that CPGs are well-suited in general and especially for distributed implementations (e.g., for snake-like or reconfigurable robots). However, there is neither a sound design methodology to solve a specific locomotor problem nor a solid theoretical foundation. In order to implement CPGs in a meaningful way, the basic gait pattern must therefore be known from the outset, which again is robot-specific. An example for the automatic generation of optimal joint-trajectories is given in Bern et al. (2019). By using a forward shooting method and an FEM-based direct kinematics simulation, high-level goals, such as forward speed or direction of movement of various soft walking robots can be met. This method does not require a priori knowledge of a motion pattern, but can not be used online without restrictions (computation time, stability, ...). However, it can be well used to find robot-specific gait patterns.

Hence, for locomotion control of a robotic platform, a robot-specific motion strategy must be known. This paper analytically derives a robot-specific mapping of desired motion (forward and rotational speed) to joint coordinates for the gecko-inspired robot from Schiller et al. (2019), which is briefly described in section 2. The mapping function is referred to as “gait law” and is presented in section 3. In section 4, the direct kinematics of the robot are approximated by a polynomial by means of simulation and experiments to allow a fast evaluation. This is necessary to implement a control strategy in section 5 that maintains the softness of the robot and allows it to approach arbitrary references in the task space. The control strategy is referred to as Gait Pattern Generator. **Figure 1** shows the systematic procedure of this paper. To summarize, the paper contributes in two ways: (i) it derives the robot-specific motion strategy for the gecko-inspired robot and (ii), for a given robot-specific motion strategy, it provides a method to control the robot's position. However, the underlying assumptions of the former can also be transferred to other soft robots, since the ability to adapt to the environment is exploited herein.



2. ROBOT AND EXPERIMENTAL SETUP

The soft robot this paper deals with has five limbs (four legs and a torso) and four feet that can be operated independently. Therefore, its joint space has nine dimensions: the five bending angles of the limbs $\alpha = [\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4]$ and the four states of fixation actuators $f = [f_0, f_1, f_2, f_3]$. Since its locomotion is only possible within two dimensions, its description in task space needs only three coordinates: the x and y position of the robot 0x and its orientation ${}^0\varepsilon$, described in the global (Cartesian) coordinate system $\{O\}$. Thus, the task space has three dimensions. A photograph of the prototype of this robot is depicted in **Figure 2A** and **Table 1** summarizes its specifications. In order to evaluate the performance of the robot, the test bench shown in **Figure 2B** was built with an embedded camera system. To measure the bending angles α , the robot orientation ε , and the robot position x , apriltags (Wang and Olson, 2016) were fixed on its body. For a more detailed description of the experimental setup, refer to the **Supplementary Material**.

3. GAIT LAW

The straight gait of the robot can be described by a single variable—the reference bending angle of the torso $\bar{\alpha}_2$. All other variables of the joint space can then be described as a function of $\bar{\alpha}_2$ by means of the gait law for the straight gait, which was derived in Seibel and Schiller (2018):

$$\bar{\alpha}_{\text{straight}} = \begin{bmatrix} \frac{\pi}{4} - \frac{\bar{\alpha}_2}{2} \\ \frac{\pi}{4} + \frac{\bar{\alpha}_2}{2} \\ \bar{\alpha}_2 \\ \frac{\pi}{4} - \frac{\bar{\alpha}_2}{2} \\ \frac{\pi}{4} + \frac{\bar{\alpha}_2}{2} \end{bmatrix}, f = \begin{bmatrix} 0 \text{ if } \bar{\alpha}_2 < 0 \text{ else } 1 \\ 1 \text{ if } \bar{\alpha}_2 < 0 \text{ else } 0 \\ 1 \text{ if } \bar{\alpha}_2 < 0 \text{ else } 0 \\ 0 \text{ if } \bar{\alpha}_2 < 0 \text{ else } 1 \end{bmatrix}. \quad (1)$$

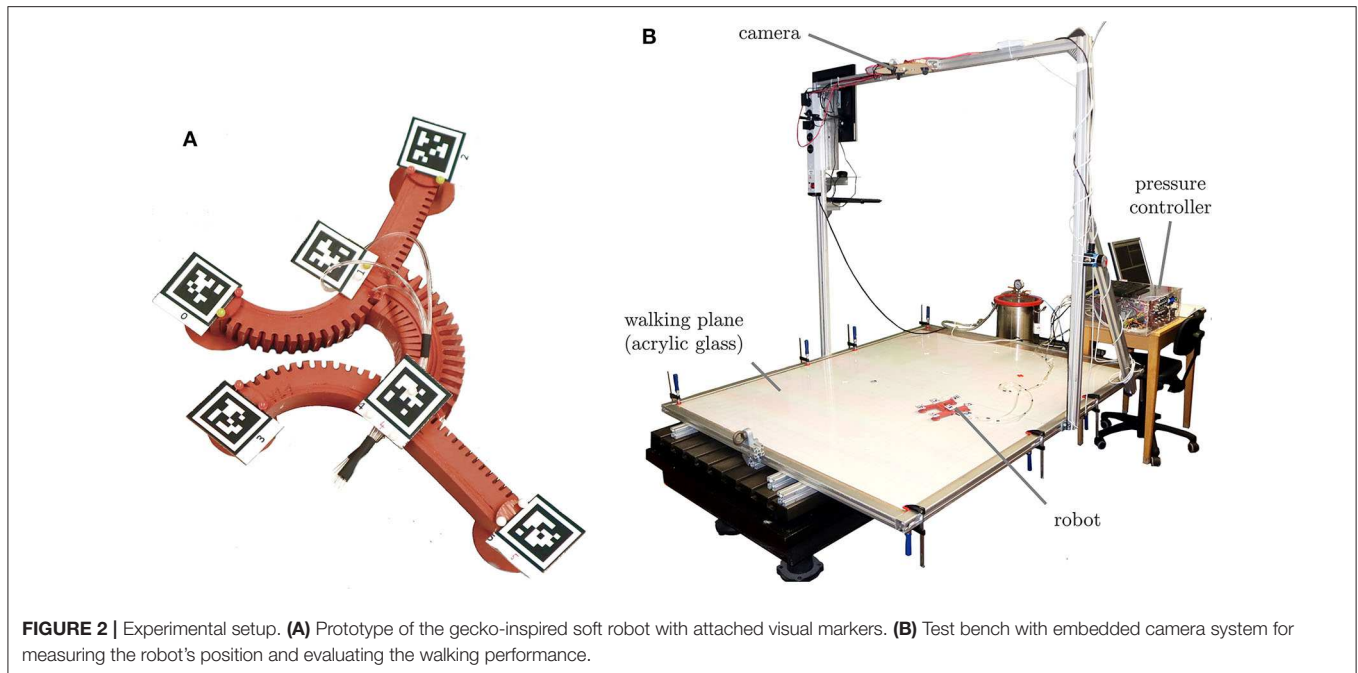


FIGURE 2 | Experimental setup. **(A)** Prototype of the gecko-inspired soft robot with attached visual markers. **(B)** Test bench with embedded camera system for measuring the robot's position and evaluating the walking performance.

TABLE 1 | Specifications of the soft robot.

Total weight	150 g
Max. speed	6 cm/s
Body length	12 cm
Body span	17 cm
Average applied pressure	0.76 bar
Pull-off force of suction cups	47 N

For a constant cycle time, the torso's bending angle is the essential measure for the forward velocity. Therefore, q_1 as the signal driving the forward velocity is introduced, and for straight gait, $\bar{\alpha}_2 = q_1$ is set. In order to operate the robot with different velocities, the angle reference $\bar{\alpha}(q_1)$ for a given step size q_1 is inverted after a certain time interval t_{move} . Hence, it jumps from $\bar{\alpha}(q_1)$ to $\bar{\alpha}(-q_1)$. The corresponding fixation reference f must also be inverted.

3.1. Derivation for General Case

The above gait law can only generate gait patterns for straight motion. It is based on the idea that the orientations of the feet always remain constant. Now, we will loosen this restriction and demand only constant orientations for the fixed feet, while the unfixed feet are allowed to rotate. This implies two cases to be considered:

1. What should be the rule for a fixed foot so that its orientation remains constant regardless of the rotation of the body?
2. What should be the rule for a free foot so that its change of orientation matches that of the body and enables a suitable initial pose for the next cycle?

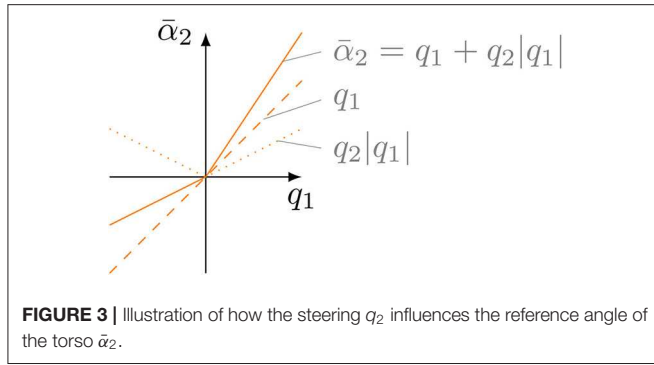
For both cases, the rules are based on the change of orientations of the feet. The orientations of feet $\varphi = [\varphi_0 \varphi_2 \varphi_3 \varphi_5]^T$ described in the global coordinate system—and consequently their change during the change of pose—can be calculated assuming constant curvature as follows:

$$\varphi(\alpha, \varepsilon) = \begin{bmatrix} \varepsilon - \frac{1}{2}\alpha_2 - \alpha_0 \\ \varepsilon - \frac{1}{2}\alpha_2 + \alpha_1 \\ \varepsilon + \frac{1}{2}\alpha_2 + \alpha_3 + \pi \\ \varepsilon + \frac{1}{2}\alpha_2 - \alpha_4 + \pi \end{bmatrix}. \quad (2)$$

Since the feet's orientations depend on the robot's orientation ε and the bending angles α , a description for the latter two is required. First, it will be discussed how to describe and how to change the walking direction of the robot ε , i.e., its orientation. From Schiller et al. (2020), it is known that the asymmetrical actuation of the torso leads to a rotation of the body. In order to describe an asymmetrical actuation, the steering factor q_2 is introduced. The reference angle for the torso $\bar{\alpha}_2$ is then described as follows:

$$\bar{\alpha}_2 = q_1 + |q_1|q_2, \quad (3)$$

where $q_2 \in [-0.5, 0.5]$ is dimensionless and shifts the reference angle of the torso $\bar{\alpha}_2$ in the direction of q_2 (compare Figure 3). In this way, the left side of the torso is actuated by $|q_1|q_2$ more in the first half of a cycle and the right side by the same amount less in the second half of the cycle. It should be noted that Equation (3) describes only one possible model for asymmetric actuation. Several models have been tested and this one has been established. Clearly, the change of orientation per cycle $\Delta\varepsilon$ is related to the steering factor q_2 and the step length q_1 . Simulation and experiment show that, for asymmetric



actuation with positive q_2 , a negative change of orientation occurs, and vice versa. The change in orientation per cycle is therefore negatively proportional to the steering factor and the step length:

$$\Delta \varepsilon \sim -|\Delta q_1| q_2,$$

where $|\Delta q_1|$ is the amount of change in torso actuation from initial pose to subsequent pose. This results in a model for orientation change per cycle $\Delta \hat{\varepsilon}$ of the body:

$$\frac{1}{2} \Delta \hat{\varepsilon} = -\tilde{c}_1 |\Delta q_1| q_2, \quad (4)$$

where the robot-specific constant \tilde{c}_1 describes the ability of the robot to rotate. Here, it is assumed that the robot rotates consistently within the cycle. Therefore, the change in orientation after a pose change (half cycle) is exactly half as much as after the entire cycle; compare to **Figure 4** where $\frac{1}{2} \Delta \varepsilon = \varepsilon_1 - \varepsilon_0 = \varepsilon_2 - \varepsilon_1$.

The second parameter for calculating the feet's orientations Equation (2) is the bending angles of the legs. Hence, a specification for the legs is needed. The structure of the straight gait law from Equation (1) was adopted for this purpose, whereby the reference angles of the legs are extended with a yet unknown term $g(q_1, q_2)$. In the following, the procedure is shown for the front left leg only (α_0). However, it can be transferred to all other legs. With this extension, the reference angle for the front left leg results in:

$$\bar{\alpha}_0 = \frac{\pi}{4} - \frac{\bar{\alpha}_2}{2} + g(q_1, q_2). \quad (5)$$

Now, the change of foot orientation when changing the pose $\Delta \varphi$ can be derived from Equations (2)–(5) by treating the references of the bending angles as the actual bending angles and assuming the body rotates according to the model from Equation (4):

$$\begin{aligned} \Delta \varphi_0 &= \varphi_{0,1} - \varphi_{0,0} \\ &= \left(\varepsilon_1 - \frac{\bar{\alpha}_{2,1}}{2} - \bar{\alpha}_{0,1} \right) - \left(\varepsilon_0 - \frac{\bar{\alpha}_{2,0}}{2} - \bar{\alpha}_{0,0} \right) \\ &= \left(\varepsilon_1 - \frac{\pi}{4} - g(q_{1,1}, q_2) \right) - \left(\varepsilon_0 - \frac{\pi}{4} - g(q_{1,0}, q_2) \right), \quad (6) \\ &= (\varepsilon_1 - \varepsilon_0) - (g(q_{1,1}, q_2) - g(q_{1,0}, q_2)) \\ &= \frac{1}{2} \Delta \hat{\varepsilon} - \Delta g(q_1, q_2) \end{aligned}$$

where $q_{1,0}$ describes the step length of the initial pose and $q_{1,1}$ that of the subsequent pose. When changing poses, the robot always jumps from $\bar{\alpha}(q_1, \cdot)$ to $\bar{\alpha}(-q_1, \cdot)$. Therefore, $q_{1,1} = -q_{1,0}$ and $g(q_{1,1}, q_2) - g(q_{1,0}, q_2)$ can be combined to $\Delta g(q_1, q_2)$. Furthermore, it is assumed that the steering factor q_2 remains unchanged when changing poses. Next, a specification for the additional term $g(q_1, q_2)$ is derived for the two cases under consideration (fixed and unfixed leg).

3.1.1. Fixed Leg

Figure 4 shows one cycle of trotting gait. Within the transition from the initial pose (black) to the middle pose (gray), the front left foot is fixed and thus its orientation should remain constant. The bending angle must be determined in such a way that the foot's orientation is kept constant, i.e., independent of q_1 or q_2 :

$$\Delta \varphi_{0,f} = \frac{1}{2} \Delta \hat{\varepsilon} - \Delta g_f(q_1, q_2) = 0 \quad \forall q_1, q_2, \quad (7)$$

where the index f denotes a fixed foot/leg. This means that the robot can change from any pose described by the general gait law to a subsequent pose without changing the orientation of its fixed feet, with the limitation that the steering factor q_2 remains constant with this change. From Equations (7) and (4), the additional term for the fixed leg results in:

$$\Delta g_f(q_1, q_2) = \frac{1}{2} \Delta \hat{\varepsilon} = -\tilde{c}_1 |\Delta q_1| q_2. \quad (8)$$

Since the sign of q_1 is always swapped when changing poses, the change of the torso actuation always results in $|\Delta q_1| = 2|q_1|$, and thus, the additional term becomes (with $c_1 = -4\tilde{c}_1$):

$$g_f(q_1, q_2) = c_1 |q_1| q_2. \quad (9)$$

Inserted in Equation (5), the reference for a fixed leg results in:

$$\alpha_{0,f} = \frac{\pi}{4} - \frac{q_1}{2} - \frac{1}{2} |q_1| q_2 + c_1 |q_1| q_2. \quad (10)$$

3.1.2. Free Leg

As the foot was previously fixed, the rotation of the body must affect its orientation in the non-fixed phase. The free foot should therefore rotate in the unfixed phase exactly as much as the body does during the entire cycle. This is illustrated in **Figure 4** where the change in orientation of the front left foot between the final pose (lightgray) and the middle pose (gray) matches exactly the rotation of the body $\Delta \hat{\varepsilon}$. With the model for the change of foot orientation from Equation (6), it must hold:

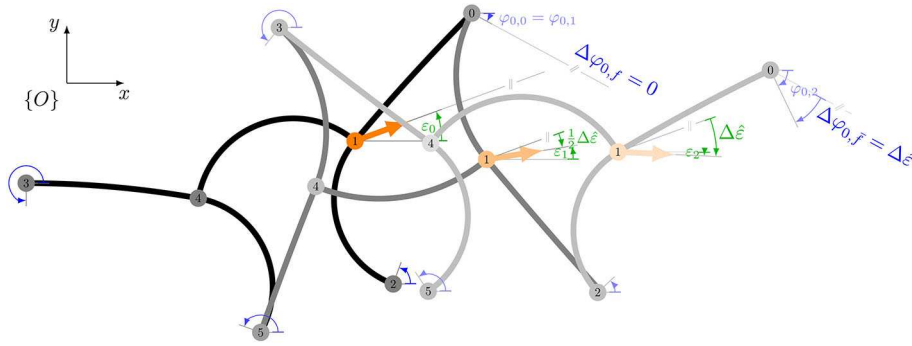


FIGURE 4 | Problem statement: which bending angles must be applied in order to turn the robot while keeping the orientation of its fixed feet? During the first change of pose, the orientation of the front left foot (φ_0) and the rear right foot (φ_5) should be kept constant. During the second change of pose, the front right foot (φ_2) and the rear left foot (φ_3) should not rotate.

$$\Delta\varphi_{0,\bar{f}} = \frac{1}{2}\Delta\hat{\varepsilon} - \Delta g_{\bar{f}}(q_1, q_2) = \Delta\hat{\varepsilon} \quad \forall q_1, q_2, \quad (11)$$

where \bar{f} indicates an unfixed foot. Clearly, this only applies if the same additional term is added again, but with swapped sign:

$$g_{\bar{f}}(q_1, q_2) = -g_f(q_1, q_2) = -c_1|q_1|q_2. \quad (12)$$

According to Equation (5), the reference for a free leg results in:

$$\alpha_{0,\bar{f}} = \frac{\pi}{4} - \frac{q_1}{2} - \frac{1}{2}|q_1|q_2 - c_1|q_1|q_2. \quad (13)$$

If a foot is fixed, we add the term $g(q_1, q_2) = c_1|q_1|q_2$ to the reference angle of the corresponding leg. If the leg is free, the additional term g is subtracted. Whether a leg is fixed or not is determined by the sign of the torso reference (see Equation 1): q_1 positive \rightarrow foot fixed, q_1 negative \rightarrow foot free. Thus, the distinction between free and fixed leg can be avoided by dropping the amount operation of q_1 in the additional term g . The sign of q_1 then automatically controls the corrective direction of the additional term g . This procedure can be performed for all legs and results in the general gait law, which is formally described as follows:

$$\bar{\alpha} = \begin{bmatrix} \frac{\pi}{4} - \frac{q_1}{2} - \frac{1}{2}|q_1|q_2 + c_1q_1q_2 \\ \frac{\pi}{4} + \frac{q_1}{2} + \frac{1}{2}|q_1|q_2 + c_1q_1q_2 \\ q_1 + |q_1|q_2 \\ \frac{\pi}{4} - \frac{q_1}{2} - \frac{1}{2}|q_1|q_2 + c_1q_1q_2 \\ \frac{\pi}{4} + \frac{q_1}{2} + \frac{1}{2}|q_1|q_2 + c_1q_1q_2 \end{bmatrix}, \quad f = (1). \quad (14)$$

The value of additional leg bending c_1 is to be determined via simulations or experiments. This is demonstrated in the **Supplementary Material** and results in $c_1 = 1$. The visualization of this law is shown in **Figure 5**. Note that the middle layer shows the special case for straight motion from Equation (1). By

introducing the index k specifying the extreme poses, references for a gait can be generated recursively by

$$\bar{\alpha}_k = \bar{\alpha}(-q_{1,k-1}, q_2), \quad f_k = \neg f_{k-1}, \quad (15)$$

where $\neg f$ is the logical negation of f .

The gait law generates reference angles for the robot, depending on step length (forward velocity) q_1 and steering factor (rotational velocity) q_2 . These two generalized coordinates define the so-called velocity space of trotting gaits, since each pair (q_1, q_2) describes another trotting gait. If q_1 and q_2 remain constant during gait, theoretically, the orientation of the fixed feet does not change. However, the derivation of this law did not examine whether the fixed feet also remain in position when switching poses. Also, the robot should have the ability to change its gait over time and should not always run the same circle with the same velocity. Therefore, q_1 and q_2 must vary. The next section examines whether this law provides useful references, despite neglecting the feet positions.

3.2. Experimental Validation

Within an experiment, it shall be analyzed whether the orientations of fixed feet actually remain constant during a cycle or ignoring the feet positions leads to significant discrepancies. The gait was slowed down ($t_{\text{move}} = 10$ s) as highly dynamic changes smear the camera images and the tags can no longer be detected by image processing. **Figure 6** shows an exemplary cycle of a gait for $q_1 = 80^\circ$ and $q_2 = -0.5$. The figure shows the mean values and standard deviations of five experiments in total. For the detailed processing steps in the evaluation, refer to the **Supplementary Material**. The upper graph shows the progression of the bending angles α and the lower graph shows the progression of the orientations φ and ε during a cycle. Initially, all feet are fixed (pose 1a). The bending angle of the front left (red line) and the rear right leg (dark blue line) differs significantly from the reference at this point in time because the robot is forced into this pose by the fixation of its feet. After about five percent of the cycle time, the front left and rear right foot are released (pose 1b). At this point, a jump in the bending

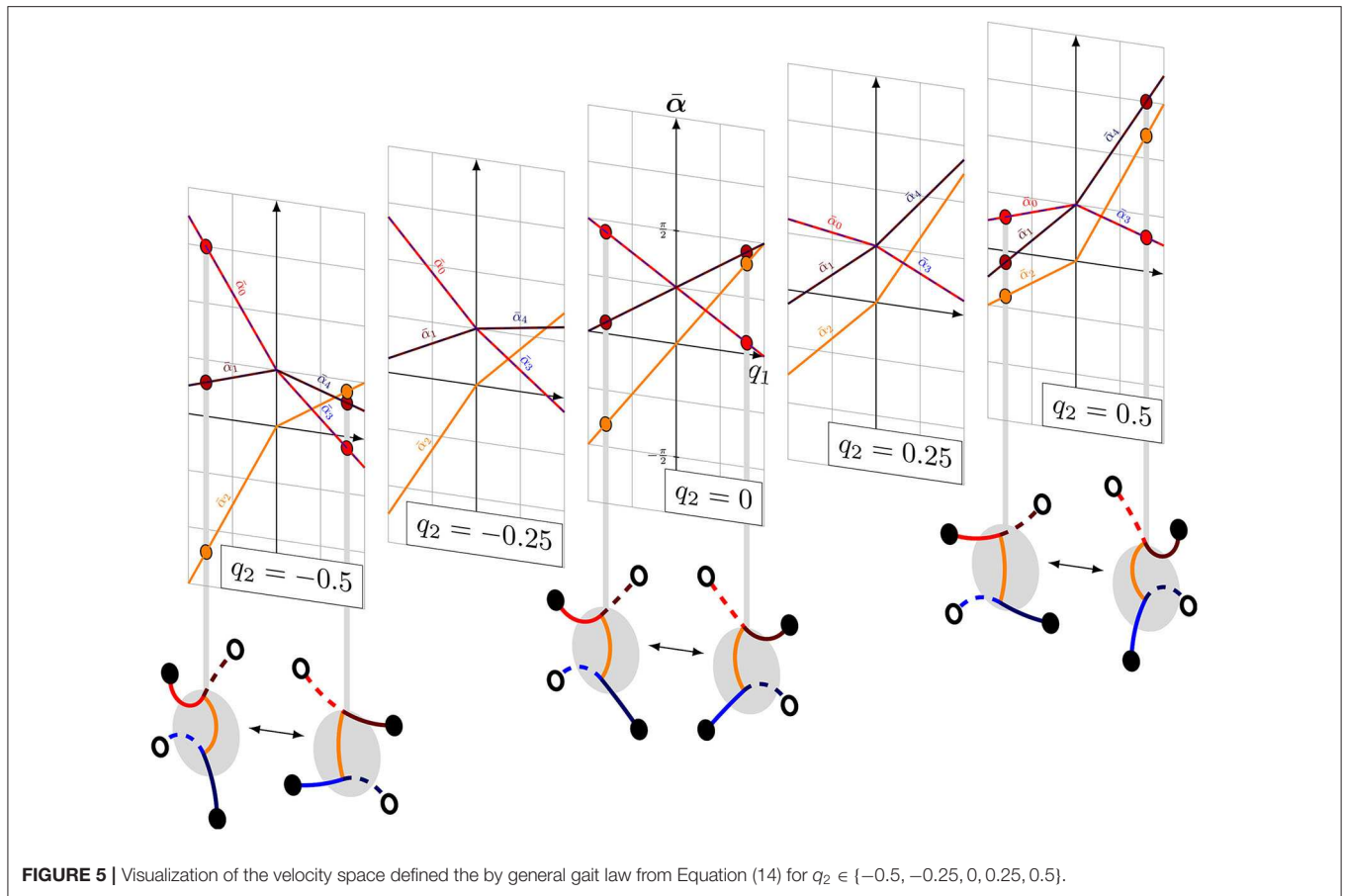


FIGURE 5 | Visualization of the velocity space defined by the general gait law from Equation (14) for $q_2 \in \{-0.5, -0.25, 0, 0.25, 0.5\}$.

angle of the two corresponding legs can be observed—the angles jump to their reference. The same effect can be observed when changing the feet fixation in the middle of the cycle (pose 2a → 2b). From this observation, it can be deduced that the robot cannot match the reference generated by the gait law because the closed kinematic chain of its parallel structure prevents it from adopting the specified bending angles. The φ graph shows that the orientations of the fixed feet remain nearly constant as assumed when deriving the gait law. An exception is the rear left foot (blue line): its orientation changes significantly during the fixed phase. As already seen in Schiller et al. (2020), the suction cups of the robot have a certain margin of rotation. This must now be utilized; otherwise, the feet would have to move (which is not possible because of the fixation). In summary, it can be concluded from the experiment in **Figure 6** that the gait law provides references which cannot be fully realized due to the closed kinematic chain, but nevertheless lead to the desired behavior.

4. APPROXIMATING THE DIRECT KINEMATICS

The next step is to determine how the robot behaves in the task space for each point (q_1, q_2) in the velocity space—that is, how it moves per cycle and by how much it rotates. Thus, the bivariate

polynomial $\Delta \mathbf{x}(q_1, q_2)$ is searched for which approximates the transformation of the velocity space into the task space (compare **Figure 1**). The form of the polynomial is defined as follows:

$$\Delta \mathbf{x}(q_1, q_2) = \begin{bmatrix} \Delta \varepsilon \\ \Delta x \\ \Delta y \end{bmatrix}, \Delta \varepsilon, \Delta x, \Delta y := \sum_{i,j} a_{i,j} q_1^i q_2^j. \quad (16)$$

In order to identify the coefficients, the velocity space is gridded and for each set of values the motion of the robot is measured. This can either be done experimentally or the simulation model is used and the movement is simulated. The result of both approaches depends on the way they are implemented. Therefore, the influencing factors must be identified and their value must be meaningfully determined. **Table 2** summarizes the conditions under which the following experiments or simulation were carried out. A detailed discussion of the experimental conditions can be found in the **Supplementary Material**.

Figure 7 shows the results of simulation (**Figure 7A**) and experiment (**Figure 7B**). In both cases, the velocity space was gridded with $q_1 \in \{50, 60, \dots, 90\}$ and $q_2 \in \{-0.5, -0.3, \dots, 0.5\}$ and a measurement was performed for each grid point. A simplified representation of the extreme poses of the resulting gait illustrates the movement. The tip of the torso of the initial pose is always at the position (q_1, q_2) and the orientation of

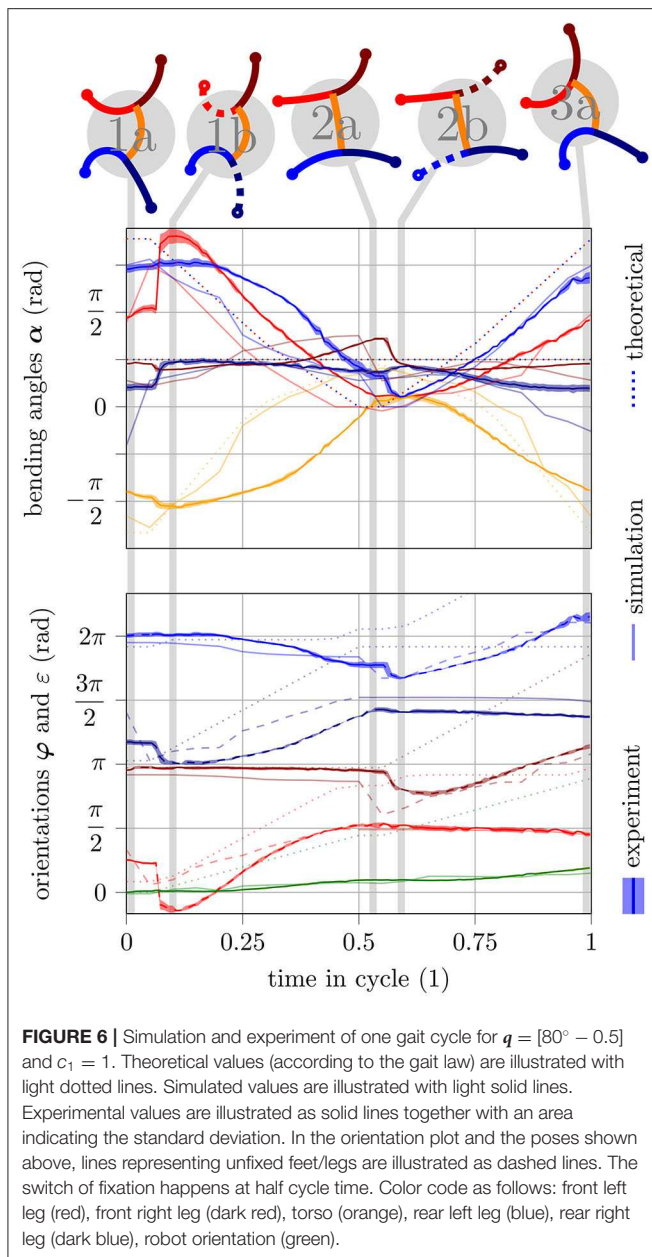


FIGURE 6 | Simulation and experiment of one gait cycle for $q = [80^\circ - 0.5]$ and $c_1 = 1$. Theoretical values (according to the gait law) are illustrated with light dotted lines. Simulated values are illustrated with light solid lines. Experimental values are illustrated as solid lines together with an area indicating the standard deviation. In the orientation plot and the poses shown above, lines representing unfixed feet/legs are illustrated as dashed lines. The switch of fixation happens at half cycle time. Color code as follows: front left leg (red), front right leg (dark red), torso (orange), rear left leg (blue), rear right leg (dark blue), robot orientation (green).

TABLE 2 | Influencing factors on simulation and experiment.

Initial pose	$\alpha_0 = \tilde{\alpha}(q_1, q_2), f_0 = [1, 0, 0, 1]$
Number of cycles	min. $n_{\text{cyc}} = 2$, drop initial cycle
Dimensions of robot	$\ell_{\text{leg}} = 9.1 \text{ cm}, \ell_{\text{torso}} = 10.3 \text{ cm}$
Weighting parameters (only simulation)	$f_l, f_o, f_a = 89, 10, 5.9$
Model order of $\Delta x(q_1, q_2)$	order = 2

the robot faces upwards. The resulting translation $[\Delta x \Delta y]^T$ per cycle is indicated by a red arrow. Besides, the orientation of the robot after a cycle is represented by a green line. The heat map in the background shows the resulting rotation $\Delta \varepsilon(q_1, q_2)$ per cycle. The numerical value of this function is noted in a

green box below the individual measurements. In the figure of the experiment (**Figure 7B**), the standard deviation of the translation is shown as a red ellipse with the corresponding semi axes. The standard deviation of the rotation is visualized as a light green triangle with an opening angle of $2\text{std}(\delta \varepsilon)$. The blue arrow shows the polynomial fit of the translation and the blue line the polynomial fit of the rotation at the corresponding grid point. A detailed view of a single experiment is shown in **Figure 8**.

In contrast to the experiment in section 3.2, in **Figure 7**, a clear deviation between simulated and experimental results can be observed. The resulting rotation and the shift in transverse direction are noticeably higher for all grid points. The simulation model does not reproduce friction effects or external disturbances, such as the influence of the supply tubes. In the previous experiment (from section 3.2), these effects played a subordinate role because of the reduced speed and the relatively short distance traveled. This experiment was executed at full speed ($t_{\text{move}} = 1 \text{ s}$); thus, friction has a significantly larger influence. Furthermore, we can observe that the experiment is not symmetrical, meaning that swapping the sign of q_2 does not yield to mirrored behavior $[\Delta x(q_1, q_2) \approx -\Delta x(q_1, -q_2)]$. This can be attributed to manufacturing inaccuracies of the robot and an optimizable pressure-bending angle calibration. The calibration procedure and associated difficulties are also discussed in the **Supplementary Material**. A final observation is that, in the experiment, the resulting rotation decreases for a large step length q_1 . This is different to the simulation, where the resulting rotation increases steadily with increasing step length. For large values of q_1 and q_2 , the gait law prescribes relatively large reference angles. If these are out of range of calibration of the respective actuator, the reference pressure is saturated to prevent damage to the robot. Exactly this effect occurs in the upper part ($q_1 \geq 80^\circ$) of **Figure 7B**. Therefore, the poses here deviate much more from their simulated counterparts in **Figure 7A** than in the lower part of the figure ($q_1 < 80^\circ$). Apart from the “over-simulation” and the missing saturation effect, the simulation reproduces the behavior very well. It can be seen as the behavior of a robot that has been perfectly manufactured and calibrated, consisting of actuators as robust as saturation is no longer necessary, whose feet have the optimum torsional stiffness, and where all friction effects have been reduced to a minimum. For the implementation of the Gait Pattern Generator, however, the actual interest focuses on the polynomial fit of the motion. In most cases, the second-order fit shown in blue matches the measurement or is at least within the standard deviation. The coefficients for the polynomial $\Delta x(q_1, q_2)$ for Equation (16) are listed in **Table 3**.

5. GAIT PATTERN GENERATOR

The last step to control the robot's position is the calculation of the optimal tuple q^* to move from the current position x closer to a given target position \bar{x} (compare **Figure 1**).

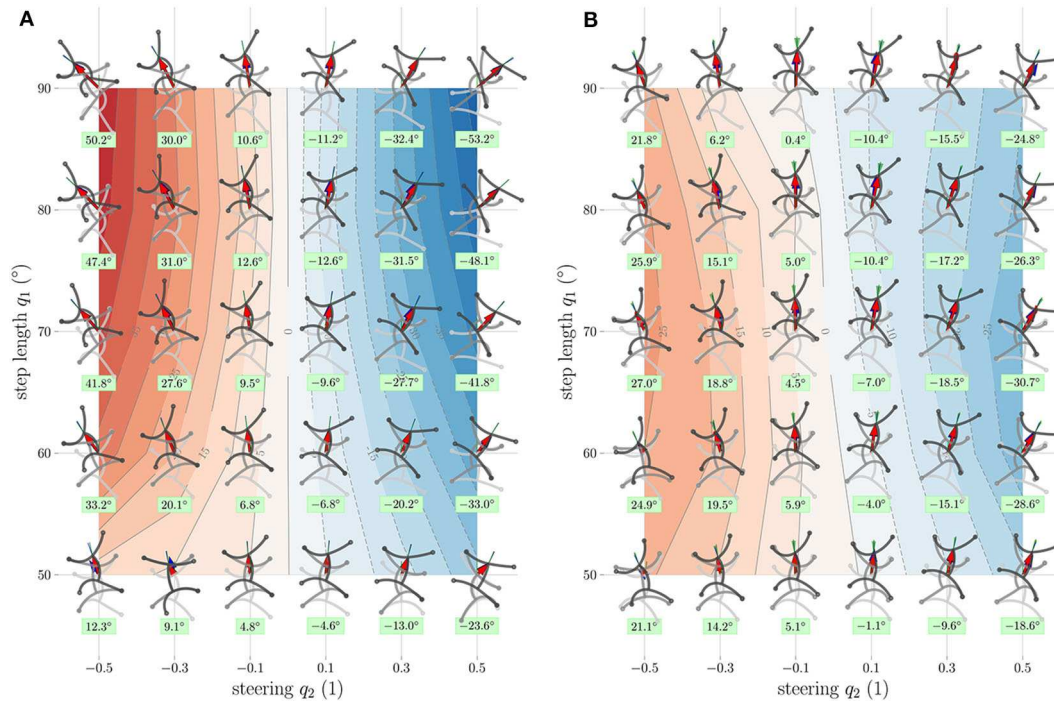


FIGURE 7 | Resulting experimental gaits according to the gait law in Equation (14) for a variation of step length and steering factor. The rows each have a constant step length q_1 and the columns a constant steering factor q_2 . Each frame shows the resulting motion of one cycle with the pattern corresponding to (q_1, q_2) . Below each frame, the rotation per cycle in degrees $\Delta\epsilon$ is stated. The heat map in the background shows the polynomial fit of $\Delta\epsilon$. The bold red vector pointing from the initial position of each individual gait to its end position is called $[\Delta x \ \Delta y]^T$. **(A)** Simulation (89, 10, 5.9) and **(B)** experiment.

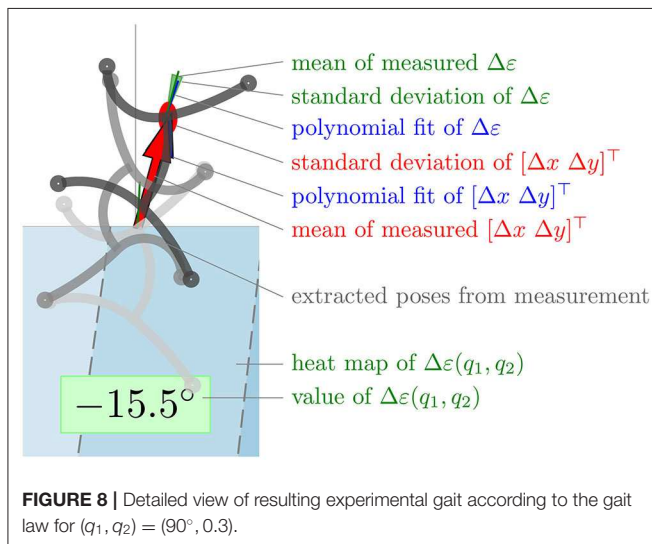


FIGURE 8 | Detailed view of resulting experimental gait according to the gait law for $(q_1, q_2) = (90^\circ, 0.3)$.

5.1. Derivation

As derived in section 4, the robot turns around $\Delta\epsilon$ and moves by $[\Delta x \ \Delta y]^T$ with each cycle. Therefore, the position of the $(n+1)$ th pose given in the coordinate system of the n th pose can be described by

$$R^{(n)} \mathbf{x}_{(n+1)} = \begin{bmatrix} \Delta x(q_1, q_2) \\ \Delta y(q_1, q_2) \end{bmatrix}, \quad (17)$$

where the index n starts from 0 indicating the initial pose and accordingly the subsequent poses. If step length q_1 and steering factor q_2 do not change during gait ($\mathbf{q} = \text{const.}$), the translation and rotation per cycle will remain the same. Let us assume that it would be possible to reach the target position in a finite number of cycles without changing the gait. Accordingly, the vector $R^{(n)} \tilde{\mathbf{x}}_{(n)}$ pointing from the n th pose to the target, can be described in the coordinate system of the n th pose as a function of the target vector of the $(n-1)$ th pose:

$$R^{(n)} \tilde{\mathbf{x}}_{(n)} = \mathbf{R}(-\Delta\epsilon) \left(R^{(n-1)} \tilde{\mathbf{x}}_{(n-1)} - R^{(n-1)} \mathbf{x}_{(n)} \right), \quad (18)$$

where $\mathbf{R} \in \mathbb{R}^{2 \times 2}$ is the rotation matrix. Since for multiple rotations around the same axis $\mathbf{R}^k(x) = \mathbf{R}(kx)$ applies, this can be formulated explicitly:

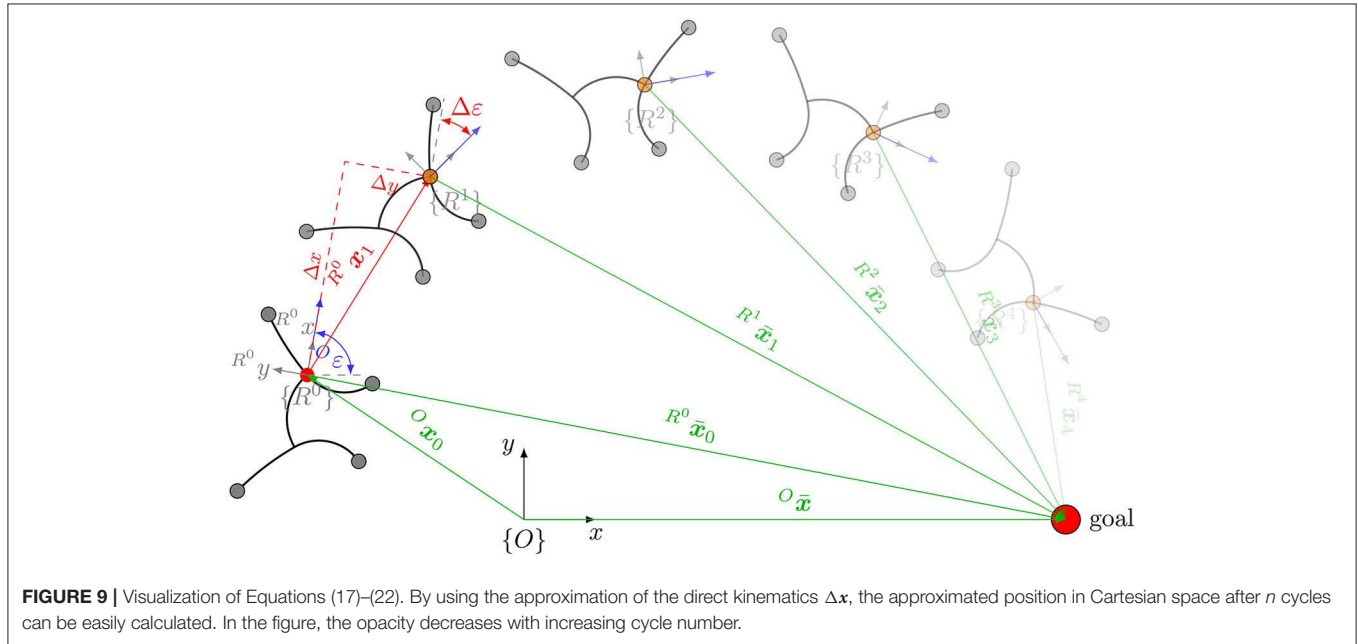
$$R^{(0)} \tilde{\mathbf{x}}_{(n)} = \mathbf{R}(-n\Delta\epsilon) R^{(0)} \tilde{\mathbf{x}}_0 - \sum_{i=0}^n \mathbf{R}(-i\Delta\epsilon) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}. \quad (19)$$

Figure 9 visualizes these formulas, whereby the opacity of poses that lie further in the future decreases. Now, the distance d_n to the target position $\tilde{\mathbf{x}}$ after n cycles of trotting with the pattern corresponding to the gait law $\tilde{\mathbf{a}}(q_1, q_2)$ can be calculated with

$$d_n(R^{(0)} \tilde{\mathbf{x}}_0, \Delta \mathbf{x}) = \left| R^{(0)} \tilde{\mathbf{x}}_{(n)} \right|_2. \quad (20)$$

TABLE 3 | Coefficients of the bivariate polynomial fit of the motion per cycle $\Delta \mathbf{x}$ for the experiment.

$f(q_1, q_2) =$	$a_{0,0}$	$a_{1,0} \cdot q_1$	$a_{0,1} \cdot q_2$	$a_{2,0} \cdot q_1^2$	$a_{0,2} \cdot q_2^2$	$a_{1,1} \cdot q_1 q_2$
$\Delta \varepsilon$ (°)	5.4154	-0.0457	-44.1944	-0.0006	0.778	-0.0832
Δx (cm)	0.1106	0.2225	11.4146	-0.0008	-17.5133	-0.1213
Δy (cm)	1.9498	-0.0682	-3.6997	0.0004	-0.0333	-0.0580



For a given target, the optimal tuple for n cycles can then be calculated as the minimum of the distance function

$$q_1^*, q_2^* = \min_{q_1, q_2 \in \mathcal{Q}} d_n^2(R^{(0)} \bar{\mathbf{x}}_0, \Delta \mathbf{x}), \quad (21)$$

where \mathcal{Q} describes the set of feasible values for q_1 and q_2 , respectively. Note that the vector $R^{(0)} \bar{\mathbf{x}}_0$ describes the target position in the coordinate system of the initial pose. In the test bed with an external camera measurement system, this vector must be calculated from the measurements of the current pose ${}^O \mathbf{x}$ and the target position ${}^O \bar{\mathbf{x}}$:

$$R^{(0)} \bar{\mathbf{x}}_0 = R(-{}^O \varepsilon) ({}^O \bar{\mathbf{x}} - {}^O \mathbf{x}). \quad (22)$$

However, the target measurement could also happen with a camera directly mounted on the robot without having to reformulate the equations, as the Gait Pattern Generator demands the target position in the robot coordinate system.

Figure 10 shows a visualization of the distance function d_n for different target points and the patterns corresponding to its minimum. In **Figure 10A**, the target is located slanted right in front of the robot and a planning horizon of $n = 1$ is considered. The minimum of the distance function is at full step length $q_1 = 90^\circ$ and a medium steering factor $q_2 = 0.3$. The resulting reference allows the robot to move precisely to the front right. In **Figure 10B**, the target is located behind the robot. With

a planning horizon of $n = 1$, the minimum distance results in the smallest allowed step length and steering. However, this solution does not bring the robot closer to the target, but it is the solution that minimizes the increase in distance. There is simply no gait pattern that can bring the robot closer to the target within only one cycle. For this reason, the planning horizon in **Figure 10C** was increased to $n = 4$. The minimum of $d_{n=4}$ is now at maximum step length and maximum steering factor for the same target position. The resulting reference leads to the desired behavior: the tightest possible right turn.

5.2. Implementation

As seen in the previous section, the distance to the target cannot always be reduced in just one cycle. The simplest strategy to solve this problem is to incrementally increase the planning horizon as long as the minimum possible distance to the target within the next n cycles $d_{n,\min}$ is larger than the current distance d_0 . Furthermore, a strategy for transitioning between different gait patterns is required. So far, all simulations and experiments have only studied the motion of consistent gaits ($\mathbf{q} = \text{const.}$). However, the pattern generator should be able to dynamically change both step length and steering factor. The easiest way to make this possible is to assume that the robot is able to switch between any gait pattern, which means to allow all possible references regardless of the current pose. Here, it is questionable whether the output \mathbf{q}^* actually minimizes the distance to the

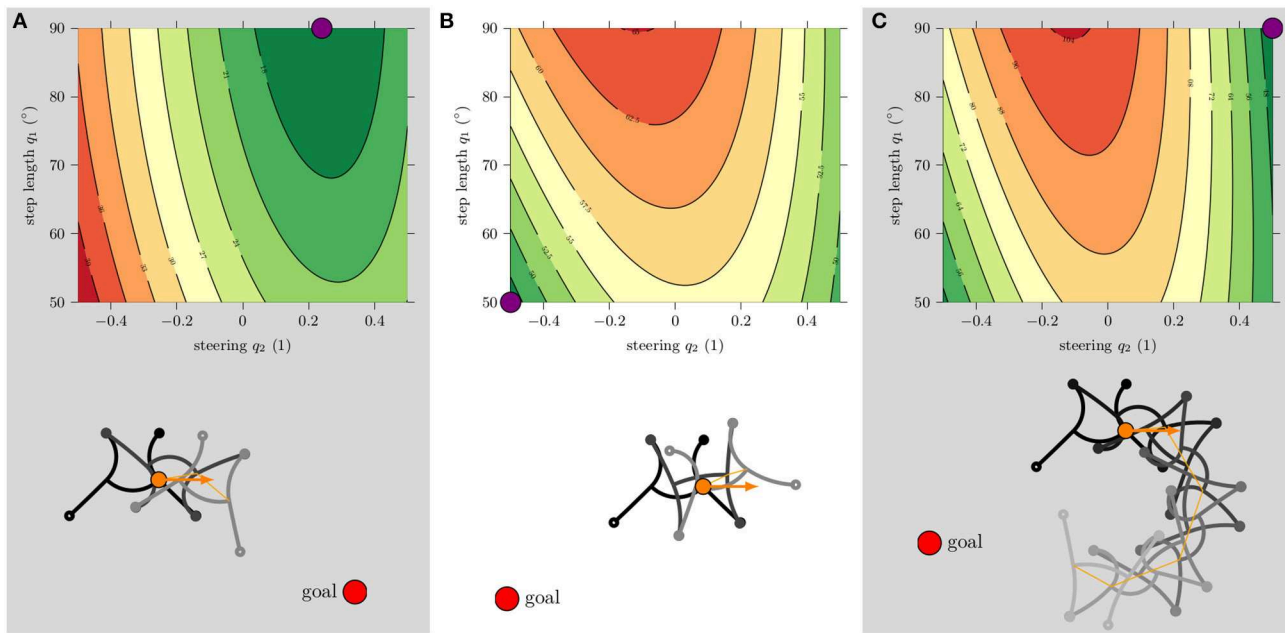


FIGURE 10 | Evaluation of the distance function d_n for different target positions \bar{x} and planning horizons n . The lowest values are represented by green and the highest values by red color. The lower image always shows the resulting simulated gait for n cycles, corresponding to the minimum distance (marked by a purple circle). Simulations were initialized with: ${}^0\mathbf{x} = (0, 0)$, ${}^0\epsilon = 0^\circ$, $\alpha_0 = [90 \ 0 \ -90 \ 90 \ 0]$, $f_0 = [1 \ 0 \ 0 \ 1]$. **(A)** Planning horizon $n = 1$ for target at $\bar{x} = (35, -20)$, **(B)** $n = 1$ for target at $\bar{x} = (-35, -20)$, and **(C)** $n = 4$ for target at $\bar{x} = (-35, -20)$.

target or whether another solution might be more suitable, since the calculation in most cases will be based on a different initial pose. Thus, it can be assumed that a different \mathbf{q}^* would be calculated when considering the current pose of the robot. But by feeding back the current position after each step and a recalculation of the reference, reaching the target position can still be ensured. Algorithm 1 implements exactly this strategy. **Figure 11A** shows the procedure as a block diagram. The sampling rate of this control loop depends on the length of half a cycle and is slightly less than 1 Hz. The Gait Pattern Generator is paused as soon as the actual distance to the target is less than a defined value $\epsilon = 5$ cm. For better comprehension, **Figure 11B** shows the low-level control architecture of the robotic system for a single actuator. Note that the simulation model mimics the coupled behavior of six of these blocks.

5.3. Experiments

In **Figure 12A**, the simulation for a list of four different target positions is shown. The next target position becomes active when $d_0 < \epsilon$ applies, i.e., the robot has almost reached the current target. In **Figure 12B**, the corresponding course of \mathbf{q} is shown. It is clear to see that both values change over time. This proves that the robot can transition between different gait patterns, at least in the simulation. The same situation is now studied in the experiment shown in **Figure 12C** where the tracks of the tags of five independent experiments are overlaid. The difference between the right and left curves is significant. While the right-hand curves have a relatively small radius, the radii of the left-hand curves are much larger. This difference has already been

Algorithm 1: Gait Pattern Generator.

```

1: procedure GENERATE REFERENCE ( $\bar{x}_k$ )
2:    $n \leftarrow 1$  ▷ start with planning horizon of 1 cycle
3:    $\mathbf{q}_k^*, d_{n,\min} \leftarrow \min_{\mathbf{q}} d_n(\bar{x}, \Delta \mathbf{x}(\mathbf{q}))$  ▷ minimal distance to goal after 1 cycle
4:   while  $d_0 > d_{n,\min}$  do ▷ stop if we get closer to goal
5:      $n \leftarrow n + 1$  ▷ increase planning horizon
6:      $\mathbf{q}_k^*, d_{n,\min} \leftarrow \min_{\mathbf{q}} d_n(\bar{x}, \Delta \mathbf{x}(\mathbf{q}))$  ▷ minimal distance to goal after  $n$  cycles
7:   end while
8:    $\mathbf{q}_{1,k}^* \leftarrow -\text{sign}(q_{1,k-1}^*)q_{1,k}^*$  ▷ switch sign of step length
9:    $\tilde{\alpha}_k \leftarrow \tilde{\alpha}(q_{1,k}^*, q_{2,k}^*)$  ▷ reference according to Eq. (14)
10:   $\mathbf{f}_k \leftarrow \neg \mathbf{f}_{k-1}$  ▷ switch fixation
11:  return  $\tilde{\alpha}_k, \mathbf{f}_k$  ▷ next reference
12: end procedure

```

noticed in the experiment from **Figure 7**; here, it is especially pronounced. The difference is due to manufacturing inaccuracies and pressure angle calibration, as discussed in section 4. The lacking ability to control the exact time of fixation of the feet also plays a role: since the strong actuation of a leg also deforms the suction cup, it may no longer be able to suck, despite negative pressure is applied. This effect is most prominent in the rear right foot. All other feet usually fix according to plan. However, the delayed fixation of the rear right leg supports a fast execution of the right turn (see **Supplementary Video**). **Figure 12D** shows the mean values and standard deviations of the step size q_1

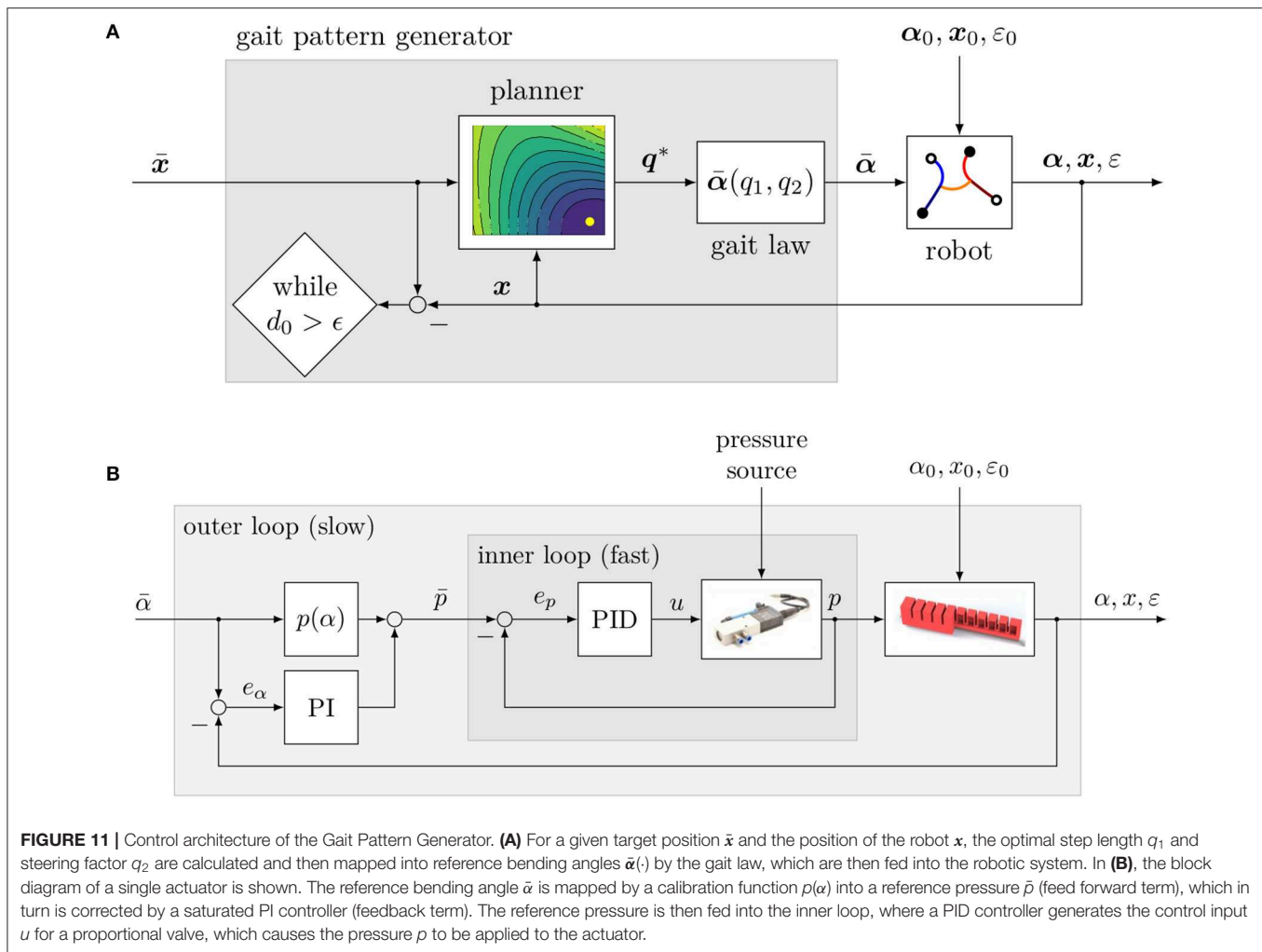
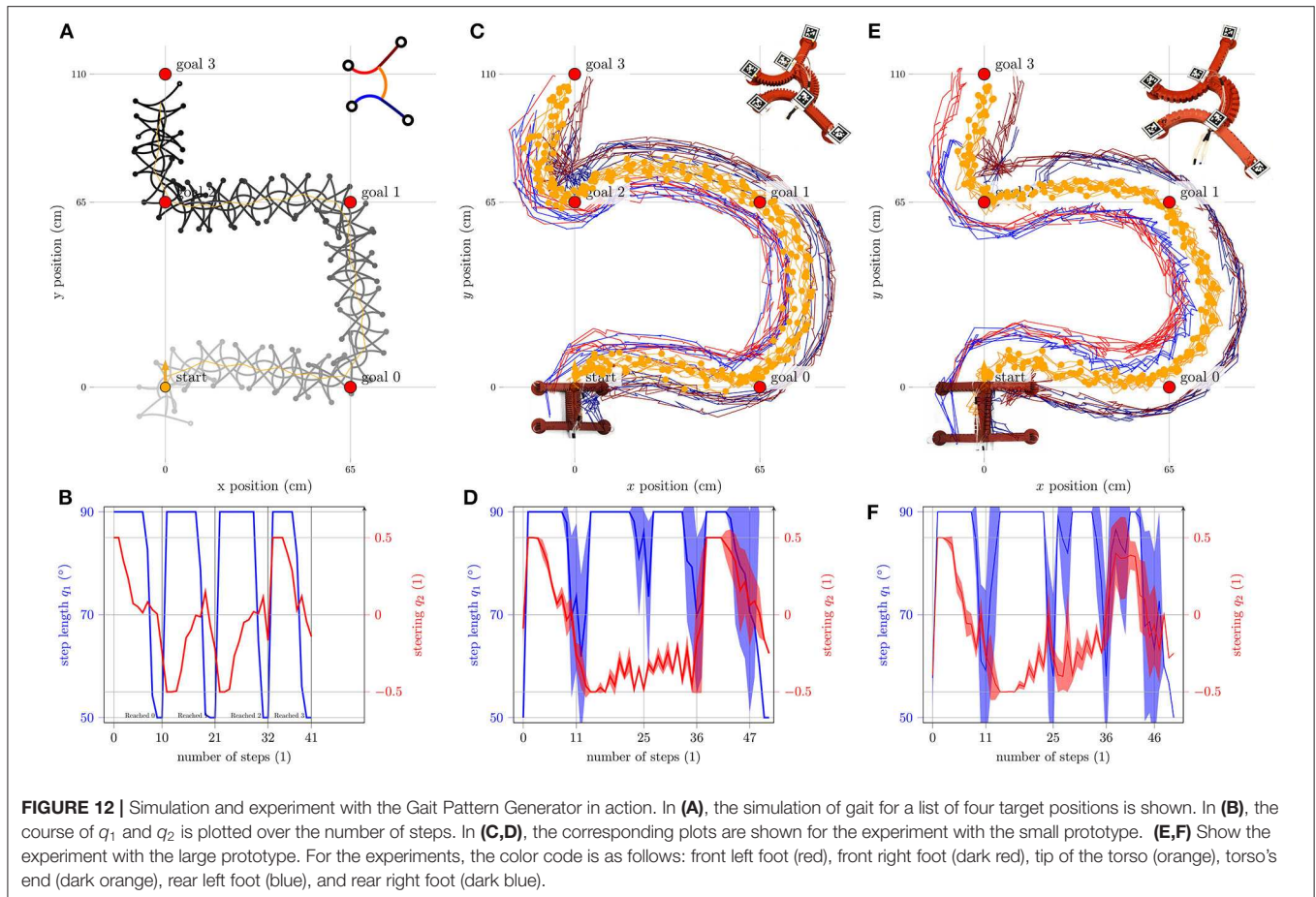


FIGURE 11 | Control architecture of the Gait Pattern Generator. **(A)** For a given target position \bar{x} and the position of the robot x , the optimal step length q_1 and steering factor q_2 are calculated and then mapped into reference bending angles $\bar{\alpha}(\cdot)$ by the gait law, which are then fed into the robotic system. In **(B)**, the block diagram of a single actuator is shown. The reference bending angle $\bar{\alpha}$ is mapped by a calibration function $p(\alpha)$ into a reference pressure \bar{p} (feed forward term), which in turn is corrected by a saturated PI controller (feedback term). The reference pressure is then fed into the inner loop, where a PID controller generates the control input u for a proportional valve, which causes the pressure p to be applied to the actuator.

(blue) and the steering factor q_2 (red). Here, the mean value was calculated over the number of steps. The different number of steps required results in a high standard deviation in the region of the four target positions. In order to reach the final target position, 45 steps were required in the fastest run and 51 steps in the worst run. The course of the mean value is similar to the simulation in **Figure 12B** and is not constant. Nevertheless, in all cases, the robot reaches the final goal and always follows a similar path. This proves that also the physical robot can transition between different gait patterns and the reproducibility of the experiments to a certain extent. **Figures 12E,F** show the results of the same experiment now performed with the robot from Seibel and Schiller (2018). The robot is basically the same, but is a little bigger (body length/span: 15/25). For the experiment, the same approximation of the direct kinematics was used (see **Table 3**), and still the robot shows the desired behavior. This shows that Δx only needs to reflect the qualitative trend. The exact values are not particularly important because as **Figures 12B,D,F** show, the step length is most of the time at the maximum and therefore the goal cannot be reached within one cycle anyway.

6. CONCLUSION

The aim of this work was position control of the gecko-inspired soft robot from Schiller et al. (2019) in Cartesian space. The solution to this complex task is based on two major simplifications: (i) the formulation of a gait law to reduce the state space of the robot from nine to two dimensions and (ii) the approximation of the direct kinematics to allow a fast evaluation. The gait law restricts the choice of possible references extremely; e.g., only specific trotting gaits are allowed. In this work, it was successfully examined whether a position control system can function with this limitations. However, it has not been investigated whether a larger permitted choice of references leads to better results. In fact, it is possible that the introduction of additional generalized coordinates or a different gait law may lead to a better performance of the robot. Furthermore, neither frictional effects nor any dynamics were considered. Also, by approximating the direct kinematics in the polynomial Δx , an assumption is made which is fulfilled only in very few cases (compare section 5.2). Instead of using the approximation, the simulation model could also be employed to find the best



possible reference for the current situation. But the simulation of one step takes an average of 0.1 s on an *AM335x 1GHz ARM® Cortex-A8* processor, which is used for control. With an average of 10 evaluations of the direct kinematics required to find the reference leading to the minimum distance, this adds up to 1 s. In contrast to a polynomial approximation where the Jacobi matrix can be easily formed to find the minimum efficiently, no analytical Jacobi matrix has been formulated for the simulation model so far. This means that when the simulation model is used, calculation would require most of the time of the cycle. However, the experiments show that the robot always reaches the target, even if the assumptions made in the derivation of the Gait Pattern Generator are not fulfilled and the approximation of the direct kinematics was done for a robot of different dimensions.

The path planning algorithm implemented is very basic, as it minimizes the Euclidean norm of the target vector, i.e., it dictates the direct path from the current position to the target. The gait law provides an intuitive way (forward and rotational speed) to control a quite complex robot and the approximation of the direct kinematics provides the resulting quantitative motion. This opens an interface to a wide variety of more dedicated path planning algorithms, as the robot can now be treated as a unicycle. For example, the path could be planned using Cartesian polynomials (Siciliano et al., 2010) and thus the robot orientation could also be controlled. Although

the softness of the robot is very complex to model, it also allows the formulation of very drastic references, even if these cannot be fulfilled at all, as hindered by the closed kinematic chain. How these contradictory demands are solved is then “computed” by the body itself. Conventional parallel kinematic robots, such as the Stewart-Gough platform, would be damaged in this case. The gecko-inspired soft robot is therefore a good example of Embodied Intelligence (Cangelosi et al., 2015) or Morphological Computation (Pfeifer and Gómez, 2009) since it does the right thing “intuitively.” This is in agreement with the principle of controlling soft robots mainly in a feed forward way in order to maintain and make use of their softness (Santina et al., 2017). The cascaded controller structure, as discussed in the introduction, can therefore also be applied to position control of mobile robots. The method of deriving a basic locomotion strategy like the presented gait law by very simple (feet rotate only in swing phase), but mathematically (with the constant curvature model) unfulfillable assumptions, can be transferred to any other soft mobile robot. Although this needs to be done individually for each robotic platform, this work can serve as a reference for future and/or existing robots.

DATA AVAILABILITY STATEMENT

The raw data generated for this study are available on request.

AUTHOR CONTRIBUTIONS

LS derived the Gait Pattern Generator, performed the experiments, and discussed the results. LS and AS wrote and revised the manuscript. AS and JS supervised the project. All authors contributed to the article and approved the submitted version.

FUNDING

The publication of this work was supported by the German Research Foundation (DFG) and Hamburg University of

Technology (TUHH) in the funding programme “Open Access Publishing.”

ACKNOWLEDGMENTS

We thank Rohat Yildiz, Duraikannan Maruthavanan, and Jakob Muchynski for the inspiration and preliminary work.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: <https://www.frontiersin.org/articles/10.3389/frobt.2020.00087/full#supplementary-material>

REFERENCES

- Bao, G., Fang, H., Chen, L., Wan, Y., Xu, F., Yang, Q., et al. (2018). Soft robotics: academic insights and perspectives through bibliometric analysis. *Soft Robot.* 5, 229–241. doi: 10.1089/soro.2017.0135
- Bern, J., Banzet, P., Poranne, R., and Coros, S. (2019). “Trajectory optimization for cable-driven soft robot locomotion,” in *Proceedings of Robotics: Science and Systems* (Freiburg im Breisgau). doi: 10.15607/RSS.2019.XV.052
- Bristow, D. A., Tharayil, M., and Alleyne, A. G. (2006). A survey of iterative learning control. *IEEE Control Syst. Mag.* 26, 96–114. doi: 10.1109/MCS.2006.1636313
- Cangelosi, A., Bongard, J., Fischer, M., and Nolfi, S. (2015). *Embodied Intelligence*. Berlin; Heidelberg: Springer. doi: 10.1007/978-3-662-43505-2_37
- Godage, I. S., Nanayakkara, T., and Caldwell, D. G. (2012). “Locomotion with continuum limbs,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Vilamoura), 293–298. doi: 10.1109/IROS.2012.6385810
- Hofer, M., and D’Andrea, R. (2018). “Design, modeling and control of a soft robotic arm,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Madrid), 1456–1463. doi: 10.1109/IROS.2018.8594221
- Hofer, M., Spannagl, L., and D’Andrea, R. (2019). “Iterative learning control for fast and accurate position tracking with an articulated soft robotic arm,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Macau), 6602–6607. doi: 10.1109/IROS40897.2019.8967636
- Horvat, T., Karakasiotis, K., Melo, K., Fleury, L., Thandiackal, R., and Ijspeert, A. J. (2015). “Inverse kinematics and reflex based controller for body-limb coordination of a salamander-like robot walking on uneven terrain,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Hamburg), 195–201. doi: 10.1109/IROS.2015.7353374
- Horvat, T., Melo, K., and Ijspeert, A. J. (2017). Spine controller for a sprawling posture robot. *IEEE Robot. Autom. Lett.* 2, 1195–1202. doi: 10.1109/LRA.2017.2664898
- Ijspeert, A. J. (2008). Central pattern generators for locomotion control in animals and robots: a review. *Neural Netw.* 21, 642–653. doi: 10.1016/j.neunet.2008.03.014
- Katzschmann, R. K., DelPreto, J., MacCurdy, R., and Rus, D. (2018). Exploration of underwater life with an acoustically controlled soft robotic fish. *Sci. Robot.* 3:eaar3449. doi: 10.1126/scirobotics.aar3449
- Lee, J., Jin, M., and Ahn, K. K. (2013). Precise tracking control of shape memory alloy actuator systems using hyperbolic tangential sliding mode control with time delay estimation. *Mechatronics* 23, 310–317. doi: 10.1016/j.mechatronics.2013.01.005
- Marchese, A. D., Komorowski, K., Onal, C. D., and Rus, D. (2014). “Design and control of a soft and continuously deformable 2D robotic manipulation system,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)* (Hong Kong), 2189–2196. doi: 10.1109/ICRA.2014.6907161
- Pfeifer, R., and Gómez, G. (2009). “Morphological computation-connecting brain, body, and environment,” in *Creating Brain-Like Intelligence*, eds B. Sendhoff, E. Körner, O. Sporns, H. Ritter and K. Doya (Berlin, Heidelberg: Springer), 66–83. doi: 10.1007/978-3-642-00616-6_5
- Pratt, J., Chew, C.-M., Torres, A., Dilworth, P., and Pratt, G. (2001). Virtual model control: an intuitive approach for bipedal locomotion. *Int. J. Robot. Res.* 20, 129–143. doi: 10.1177/02783640122067309
- Qin, L., Liang, X., Huang, H., Chui, C. K., Yeow, R. C.-H., and Zhu, J. (2019). A versatile soft crawling robot with rapid locomotion. *Soft Robot.* 6, 455–467. doi: 10.1089/soro.2018.0124
- Rus, D., and Tolley, M. T. (2015). Design, fabrication and control of soft robots. *Nature* 521, 467–475. doi: 10.1038/nature14543
- Santina, C. D., Bianchi, M., Grioli, G., Angelini, F., Catalano, M., Garabini, M., et al. (2017). Controlling soft robots: balancing feedback and feedforward elements. *IEEE Robot. Autom. Mag.* 24, 75–83. doi: 10.1109/MRA.2016.2636360
- Santina, C. D., Katzschmann, R. K., Bicchi, A., and Rus, D. (2020). Model-based dynamic feedback control of a planar soft robot: trajectory tracking and interaction with the environment. *Int. J. Robot. Res.* 39, 490–513. doi: 10.1177/0278364919897292
- Schiller, L., Seibel, A., and Schlattmann, J. (2019). Toward a gecko-inspired, climbing soft robot. *Front. Neurobot.* 13:106. doi: 10.3389/fnbot.2019.00106
- Schiller, L., Seibel, A., and Schlattmann, J. (2020). A lightweight simulation model for soft robot’s locomotion and its application to trajectory optimization. *IEEE Robot. Autom. Lett.* 5, 1199–1206. doi: 10.1109/LRA.2020.2966396
- Seibel, A., and Schiller, L. (2018). Systematic engineering design helps creating new soft machines. *Robot. Biomimet.* 5. doi: 10.1186/s40638-018-0088-4
- Shepherd, R. F., Ilievski, F., Choi, W., Morin, S. A., Stokes, A. A., Mazzeo, A. D., et al. (2011). Multigait soft robot. *Proc. Natl. Acad. Sci. U.S.A.* 108, 20400–20403. doi: 10.1073/pnas.1116564108
- Siciliano, B., Sciacicco, L., Villani, L., and Oriolo, G. (2010). *Robotics: Modelling, Planning and Control*. London: Springer. doi: 10.1007/978-1-84628-642-1
- Tolley, M. T., Shepherd, R. F., Mosadegh, B., Galloway, K. C., Wehner, M., Karpelson, M., et al. (2014). A resilient, untethered soft robot. *Soft Robot.* 1, 213–223. doi: 10.1089/soro.2014.0008
- Wang, J., and Olson, E. (2016). “Apriltag 2: Efficient and robust fiducial detection,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Daejeon), 4193–4198. doi: 10.1109/IROS.2016.7759617
- Zhang, W., and Polygerinos, P. (2018). “Distributed planning of multi-segment soft robotic arms,” in *2018 Annual American Control Conference (ACC)* (Milwaukee, WI), 2096–2101. doi: 10.23919/ACC.2018.8430682

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2020 Schiller, Seibel and Schlattmann. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.