Hindawi
Complexity
Volume 2019, Article ID 2712015, 12 pages
https://doi.org/10.1155/2019/2712015

Research Article

Natural Growth-Inspired Distributed Self-Reconfiguration of UBot Robots

Dongyang Bie[1,2] **Iqbal Sajid**[3] **Jianda Han,**[1] **Jie Zhao**[4] **and Yanhe Zhu**[4]

[1]*The Institute of Robotics and Automatic Information Systems, College of Computer and Control Engineering, and the Tianjin Key Laboratory of Intelligent Robotics, Nankai University, China*
[2]*State key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, China*
[3]*Department of Mechatronics & Control Engineering, University of Engineering and Technology (UET), Lahore 54890, Pakistan*
[4]*State Key Laboratory of Robotics and System (HIT), Harbin Institute of Technology, China*

Correspondence should be addressed to Yanhe Zhu; yhzhu@hit.edu.cn

Received 5 July 2018; Accepted 16 September 2018; Published 2 January 2019

Guest Editor: Pieter Simoens

The decentralized self-reconfiguration of modular robots has been a challenging problem. This work proposed a biological method inspired by the plant growth for the distributed self-reconfiguration of UBot systems. L-systems are implemented to construct target topology, and turtle interpretation is extended to lead the self-reconfiguration process. Parametric reproduction rules introduce the external influence to the reconfiguration process by distributed modules' local sensing. Each module can move independently to change relative positions, and robotic structures develop in the natural growth style. This leads to a convergent and environmentally sensitive control method for the distributed self-reconfiguration. Reconfiguration processes can converge to desired configuration and are scalable to module numbers by reproducing predefined substructures in principle. The overall performance of the proposed strategy is evaluated with simulations and 11 experiments. Simulation and experimental results turn out to be convergent and environmentally sensitive.

1. Introduction

Modular self-reconfigurable (MSR) robotic systems [1, 2] consist of multi-independent working modules. Each module is a complete robot with onboard battery, sensing, computing, and moving ability. The most central activity of a MSR robot is to change the global morphology through the relative motion of inner modules, which is also known as self-reconfiguration. MSR robots provide a unique advantage over traditional robotic technologies in terms of reconfigurability.

While the centralized mechanism for optimal control having been proven to be a NP problem [3, 4], the decentralized approaches have been studied from different points of view and are still on the focus in order to achieve effective solutions [5]. From the decentralized nature by multi-independent agent point of view, the self-reconfiguration process of MSR robots is a complex and collective decision-making system. The behavior of the individual modules formulates the overall state of the global robotic system. Decentralized modular self-reconfigurable robotic systems are generally analyzed by studying the transition of relative positions of inner modules and their states, referred to local surroundings including the neighboring module environments. All the local conditions of multi-independent modules contribute to the robotic system's global states.

The state space of whole robots evolves over time along the relative movement of local modules. With neighboring modules having multiple relative connecting orientations, the global robot is a nonlinear complex system. By the local motion of independent modules, robots can transfer to completely new states, which is also the self-reconfiguration of modular self-reconfigurable robots. Choosing an appropriate mechanism for controlling the self-reconfiguration is an extremely hard task [6–10], especially when considering the environmental influence [11–14].

To solve this challenging problem of MSR robotics, principles that govern self-organization of natural systems are translated into bioinspired algorithms that can exhibit

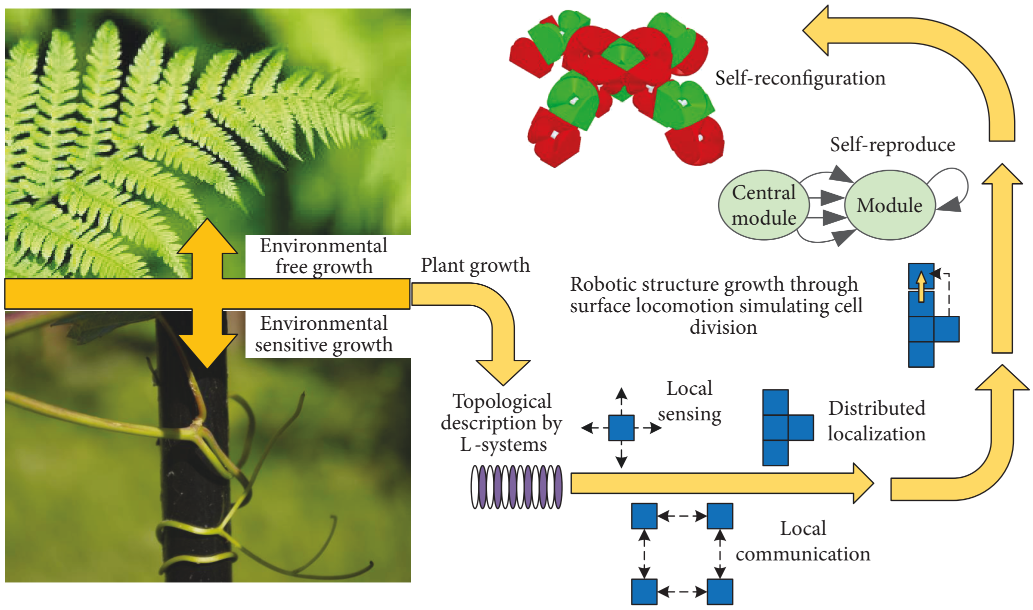

Figure 1: Plant growth-inspired self-reconfiguration.

comparable self-organizing behaviors in MSR robots. Bioinspired approaches take their motivation from the self-organization property of multicellular organisms, such as abstraction methods [15–17] and solutions methods [18–22] proposed by Ahmadzadeh and Masehian [23]. We can also cite methods based on cellular automata (CA) [24, 25] and cell morphogenesis [26].

The used bioinspired methods originate from biological self-organizing systems and aim at realizing self-reconfiguration through emergence. This emergent behavior is realized through a bottom-up process. There is no central commander in this process. The global behaviors generate through simple local interactions between individuals in the absence of a central commander. As common to most emergent processes, the convergent problem [27] is also a cornerstone in developing a bioinspired method for decentralized self-reconfiguration of MSR robots.

While both the robustness to module fails and the scalability to module numbers have roots in local and distributed nature of algorithms inspired from biological systems [28], another impediment on the way of developing bioinspired methods for self-reconfiguration is the fact that robotic systems lack the self-reproduction ability, which is crucial in biological organisms and their underlying mechanisms [15].

In this paper, a decentralized control method inspired from plant growth is designed for the decentralized self-reconfiguration of UBot robots. Plants can converge to species phenotype and respond to external touch (or thigmotropism) of an object, as shown in Figure 1. When defining the external touch as environmental influence, plant growth can be divided as environmental free growth and environmentally sensitive growth. In this work, both the plant growth and its response to touch inspire the control of the environmental free reconfiguration and environmentally sensitive reconfiguration of UBot robots.

For the high ability to simulate plant growth with simple symbols and rewriting rules, Lindenmayer systems (L-systems) [29, 30] are introduced to model the self-reconfiguration process of UBot robots. L-systems can provide topological description of target configuration for robot systems, which is similar to biological DNA. Depending on local sensing and independent motion control along gradient attraction [24], local modules can move on the surface of other modules to new positions. This surface locomotion can simulate the self-reproduction of cells. The decentralized localization problem [28, 31] is solved by using a relative location between independent modules.

The proposed method represents a novelty in the framework of self-reconfigurable robots in two ways. This paper proposes a symbolic representation of mechanical robot topology beyond its physical representation. This representation opens a door for theoretical studies related to formal languages and more complex representation which deserves a further studies. Moreover, the chosen symbol representation has a biological inspiration and bridges the area of MSR robots with plant modeling field. This bridge can allow the flow of ideas, problems and solutions enriching both research areas.

The proposed method is inspired from natural growth and distributed nature of modular robots. We would like to remark the convergence to desired topology and self-adaption to touching objects during self-reconfiguration process. Self-reconfiguration by the proposed solution is convergent to predefined configurations and scalable with the number of working modules by repeating desired segments in predefined relative orientations. The reconfiguration is also environmentally sensitive by self-adapting to touching object, just like climbing plants. For the sake of simplicity, we present some examples with simple configurations in the simulation section. Obviously, more complex structures need more modules, but the principles of the decentralized representation are the same.

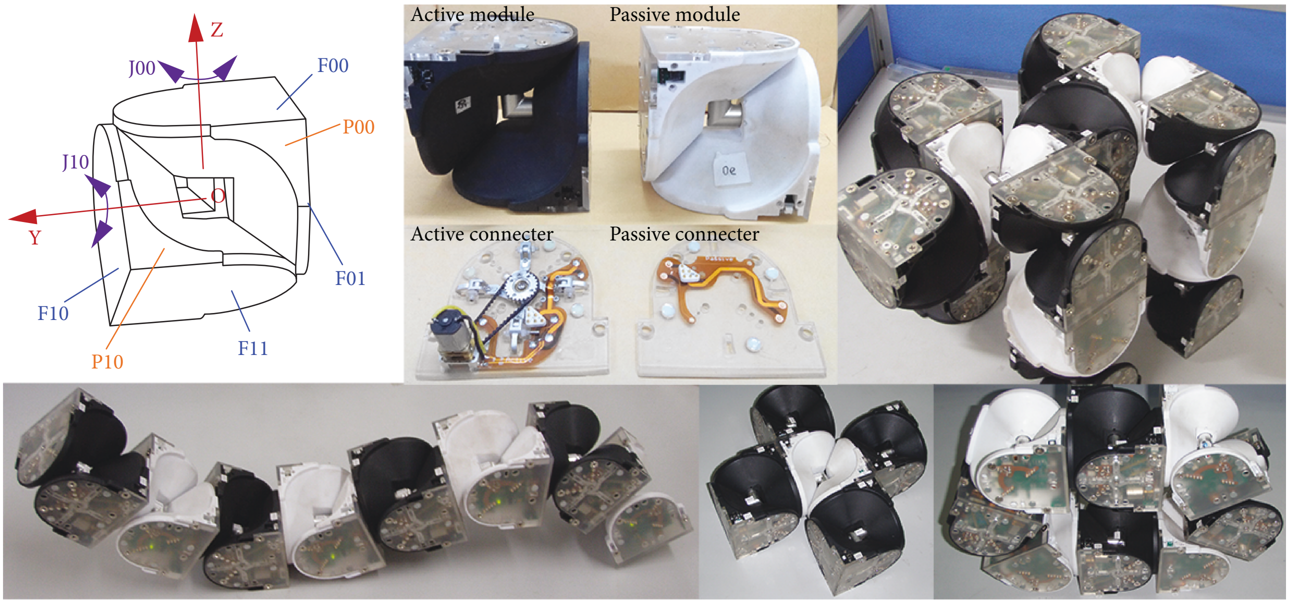

Figure 2: UBot modules and some typical robotic configurations.

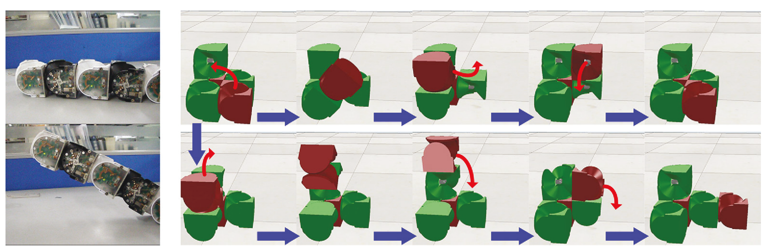

Figure 3: The lifting ability of UBot modules and their relative motion ability.

The paper is organized as follows. Firstly, the structure of UBot modular robot and its sensing support for kinematics motion are described in Section 2. In Section 3, we recall some basics on L-systems and show how the turtle interpretation of L-systems can be extended to handle reconfiguration process. And the overall performance of the proposed strategy is evaluated by simulations in Section 4 and experiments in Section 5. Finally, this paper is concluded in Section 6.

2. UBot Systems

2.1. Mechanical Structure and Sensing Ability. UBot system is a hybrid-type MSR robot developed in our laboratory, as shown in Figure 2. With an 80 mm × 80 mm × 80 mm lattice shape, each module contains two perpendicular rotating joints ($J00$ and $J10$) connecting two body parts ($P00$ and $P10$), each of which has two connecting faces ($F00$ and $F01$ on $P00$ and $F10$ and $F11$ on $P10$). UBot modules are designed as active modules and passive modules. Active modules can connect to the neighboring passive module through inner hook-type connecting mechanism. Through the various connecting ways, UBot modules can be connected to multirobotic configurations, such as branching structure, snake shape, flag shape, and rolling shape. More details about mechanical information of UBot modules can be found in our existing work [32].

2.2. Motion Mode. UBot modules can change relative positions by the assistance of neighboring modules. Figure 3 shows the lattice motion process of five UBot modules. The red module flexibly moves to neighboring lattices with the assistance of neighboring green modules. One mechanical UBot module can lift at the most four modules in linear connection. This capacity provides reliable loading ability for independent movement. And the design of perpendicular joints provides UBot modules the flexible movement ability.

2.3. Robotic Growth by Surface Locomotion. The surface locomotion can simulate the natural growth by continually moving modules to the front of global movement. The lattice construction and motion style of UBot modules leads to the rule-based control. UBot modules can move around on the surface of other modules by the implementation of CA. Figure 4 shows the CA rules for UBot system and one locomotion through narrow space. More details about the distributed surface locomotion control of UBot robots were given in

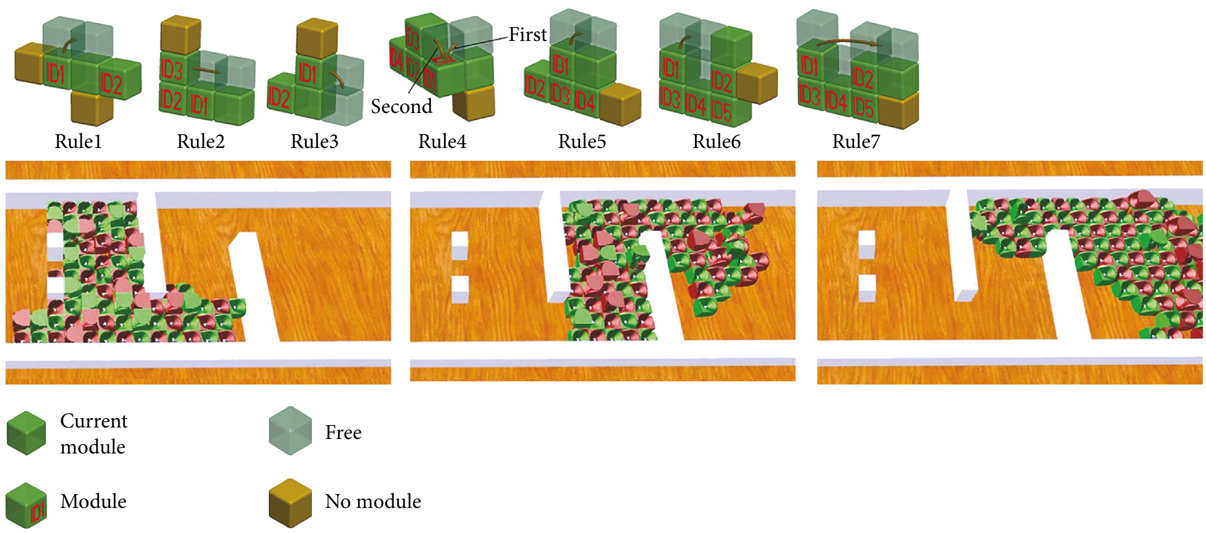

FIGURE 4: Surface locomotion of UBot robots controlled by CA.

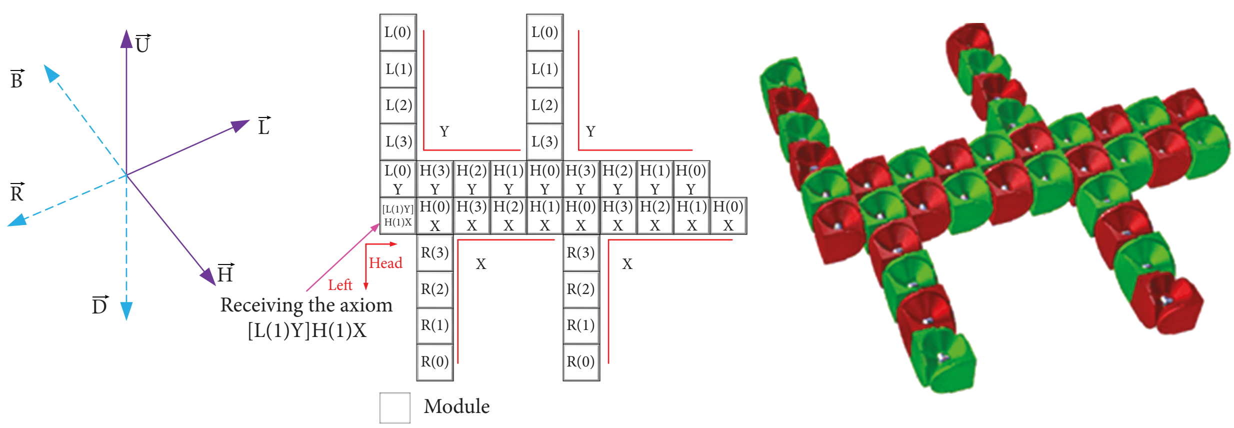

FIGURE 5: The self-similar character of lattice configuration by the L-system in (2).

our published work [24]. This surface locomotion contributes to the robotic growth of global reconfiguration.

3. Materials and Methods

3.1. L-Systems and Geometry Interpretation. L-systems have been successfully used for creating complex structures due to its expression power for complex phenomenon and the existence of algorithms for its automatic generation [33–35].

The central concept of L-systems is rewriting, which is a technique for defining complex objects by successively replacing parts of a simple initial object using a set of rewriting rules or productions. Equation (1) shows the format of rewriting rules consisting of two items—the predecessor and the successor. In applying production rules specified in the L-system, the predecessor will be recursively expanded and replaced with the relevant successors [36, 37].

$$\text{Predecessor} \longrightarrow \text{successor}. \tag{1}$$

As robotic structures exist in 3D lattice space, L-systems are extended to 3D lattice space with heading $(\vec{H})$, left $(\vec{L})$, and up $(\vec{U})$ directions for coordinate directions, as shown in Figure 5. To simplify expression complexity, the opposite directions are also taken for L-system symbols, which are back $(\vec{B})$, right $(\vec{R})$, and down $(\vec{D})$. Multiparameter properties can also be added to those extended L-system symbols. This work uses the length parameter for robotic structures in different lengths. Those properties can help organize robotic topology in both local predictions for decentralized modules and the relative position in global state. For example, the symbol $R(l)$ describes a segment with l modules linearly connected in the right direction.

Consider the L-system in (2), which consists of two production rules with processor symbols X and Y. Taking rule 1 as an example, it means that each occurrence of X will be replaced by the precise successor string of symbols ("$[R(4)]H(4)X$"). The symbols "X" and "Y" append new symbols through rewriting in the same way. The rewriting process can expend a string with circulatory symbols, as

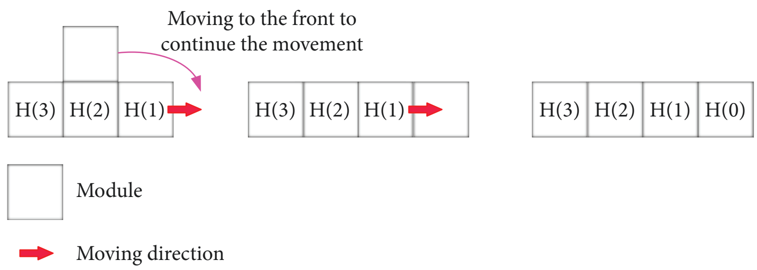

Figure 6: Robotic structures develop by attaching new modules at the growing direction.

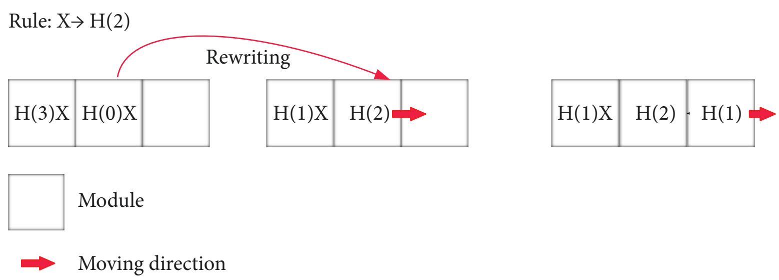

Figure 7: Decentralized rewriting work by independent modules.

symbols in the successor of rewriting rules are repeated in each rewriting work.

$$\begin{aligned} &\text{L-systems}: \\ &\quad \text{Axiom}: \quad [L(1)Y]H(1)X \\ &\quad \text{Rule 1}: \quad X \longrightarrow [R(4)]H(4)X \\ &\quad \text{Rule 2}: \quad Y \longrightarrow [L(4)]H(4)Y \end{aligned} \tag{2}$$

L-systems are conceived as a mathematical theory of development. Several geometric interpretations of L-systems have been proposed for turning them into a versatile tool for fractal and plant modeling. One of those graphic interpretation methods is described by Prusinkiewicz [38] and Prusinkiewicz and Lindenmayer [36] and later used widely, called turtle interpretation.

For implementation reasons, turtle interpretations are extended to lead the self-reconfiguration process of modular robots [28, 39]. The module with L-system symbols, called turtle module, does the moving search work as turtles in graphical interpretation and attaches new modules at neighboring lattice as L-systems described. The surface locomotion by CA [24] can simulate cell self-reproduction by moving new modules to the growing front. As shown in Figure 6, when the front lattice on segment growing direction has no module, the turtle module will generate gradient information [25] to attract mobile modules moving to the position. New attached modules will change to turtle modules and receive L-systems for the following interpretation. In this way, MSR robots reconfigure in the development style by continually moving modules to the growing front.

L-system symbols are translated between turtle modules along the growing direction. The length parameter l (in L-system symbols $H(l)$) is decreased by one in each translation from one module to the neighboring module at the growing direction. Robotic structures keep growing until the length parameter decreases to zero ($l = 0$). For example, in the decreasing process from $H(3)$ to $H(2)$ and then $H(1)$ in Figure 6, the development stops at the module with symbol $H(0)$.

The rewriting work is also managed by decentralized modules. Turtle modules keep interpreting the first symbol of received L-system symbols. When the symbol matches any reproduction rules, it will be replaced by the successor symbols and then the first symbol in the new string will be interpreted by this module. Interpreted symbols, with length parameter $l = 0$, are deleted, and the following symbols start being interpreted. For example, in the interpretation in Figure 7, the module with symbol $H(0)X$ deletes the interpreted symbols $H(0)$ and then starts interpreting symbol X.

Instead of a global position of each module, a relative strategy for decentralized localization is used. Through development in growth style by the turtle interpretation, a module gets its local position with respect to the connected turtle module. The new attached modules need not know its global position in the whole system. It just connects to the turtle module in the needed direction, and then it gets the needed position by the whole system.

As shown in Figure 5, a lattice structure corresponds to the L-system in (2). The interpretation process starts from the module that receives the axiom $[L(1)Y]H(1)X$. The resulting branching structure consists of substructures in predefined topology and is connected in predefined relative orientation by rule 1 and rule 2.

From module state point of view, modules have three kinds of interpretation state during the self-reconfiguration driven by L-systems. As shown in Figure 8, all modules are initially in mobile state. During the turtle interpretation, the module with a turtle in is the turtle module, which does the moving search work by interpreting L-system symbols as

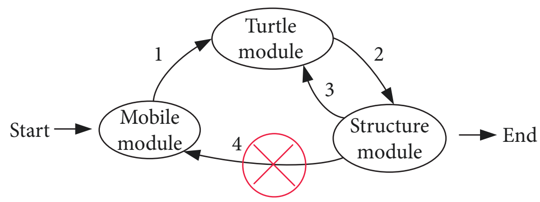

Figure 8: State translation of modules during self-reconfiguration.

turtles in computer graphics. The turtle module translates L-system symbols to the neighboring module, which is on the lattice position that the turtle will move in. In condition 1, mobile modules receiving L-system symbols change state to turtle module to continue the following interpretation. In condition 2, the turtle module changes to a structure module, when all neighboring situations satisfy the symbol description by inner L-system symbols. In this paper, we focus on the forward development as plant growth. So structure modules will be part of self-reconfiguration result and will maintain the current position without no longer movement, which is why condition 4 does not stand. But structure modules may turn back to turtle modules when receiving L-system symbols, which is condition 3 in Figure 8.

The resultant string in each generation of the L-system in (2) is deterministic by the axiom and reproduction rules, because no external influence is considered during the rewriting process. This class of L-systems is termed D0L-systems [37], pronounced dee-zero-ell-systems, which are both deterministic and context-free. D0 stands for deterministic and 0-context or context-free. We use D0L-systems for the convergent self-reconfiguration of UBot system with predefined target structures by L-systems.

3.2. Parametric L-System for Environmentally Sensitive Growth. Besides the deterministic construction by D0L-systems, parametric L-systems [40] can describe plant growth process with environmental interaction. Parametric L-systems make use of the numerical attributes of modules in the selection of an appropriate production. As shown in (3), parametric production rules are the ones in which the left-hand side has a parameter and the newly created tokens that are on the right-hand side have parameters depending on the parameter of the left-hand side.

$$\text{Predecessor}|\text{condition} \longrightarrow \text{successor} \tag{3}$$

When using parametric L-systems, configurations are discretized into strings of particles where distributed modules handle the current growth depending on local condition. The behavior of a turtle module is determined by its internal state and external conditions. The internal state refers to the current values of containing L-system symbols. For modeling response to external touch of plant growth, modules take the distance to the surface of objects as their external conditions.

As the touch response of plant growth shows climbing character, two main factors influence the emergent process of robotic reconfiguration: collision avoidance and separation avoidance. L-system symbols are extended to have two distance parameters (d_f, d_l) to manage the collision and separation problem. The symbol $F_{(d_f,d_l)}$ has growing direction F, front distance d_f, and lateral distance d_l to the surface of objects. To avoid collision with external object, the developing front must be absent from the surface at least one module's width (W_{UBot}), then we have the relationship: $d_f >= W_{\text{UBot}}$. As shown in Figure 9, to avoid separation from target object, the development must be close enough to the surface. The lateral distance (d_l)must be less than one module's width, then we have the relationship for separation avoidance: $d_l < W_{UBot}$.

For *collision detection and avoidance*, the turtle module avoids collision by changing growing direction using rule 1 in (4). After the collision avoidance, segments grow in a new direction that is parallel with the surface of the target object $F_{(d_f,d_l)}(d_f > W_{\text{UBot}})$. For example, the turtle module changes growing direction from F_{d_1} to F_{d_2} for collision avoidance in Figure 9.

$$\text{Rule 1}: F_{(d_f,d_l)}|d_f < W_{\text{UBot}} \longrightarrow F_{(d_f,d_l)}\left(d_f >= W_{\text{UBot}}\right). \tag{4}$$

For *separation detection and avoidance*, robotic segments need to develop along the surface of the target object. So the lateral distance (d_l) must not be too large. When the distance $d_l >= W_{\text{UBot}}$, turtle modules change growing direction using rule 2 in (5). For example, the turtle module changes growing direction from F_{d_3} to F_{d_4} for separation avoidance in Figure 9.

$$\text{Rule 2}: F_{(d_f,d_l)}|d_l >= W_{\text{UBot}} \longrightarrow F_{(d_f,d_l)}(d_l < W_{\text{UBot}}). \tag{5}$$

In this touch-sensitive reconfiguration process, the length parameter is not limited. It is assumed that a robot has enough modules for structure development. Both collision avoidance (rule 1) and separation avoidance (rule 2) change the growing direction to keep robotic structures developing along the surface of objects.

4. Simulation Results

4.1. Convergent and Scalable Reconfiguration in Open Space. The convergence problem is to derive the self-reconfiguration process to predefined topology. This is also an open problem in bioinspired control methods for decentralized robotic systems. While using L-systems constructing target configuration, the sequential movement of turtle interpretation guarantees robotic structures developing continuously. Local communication and relative orientation detection provide reliable translation of L-system symbols and turtle coordinate. So the proposed method in this article is convergent theoretically.

The D0L-system in (2) is taken to illustrate the convergence in simulations. As shown in Figure 10, a robot system consisting of 36 UBot modules, with the axiom and reproduction rules recorded in every module, reconfigures to a branching structure automatically. When the robotic

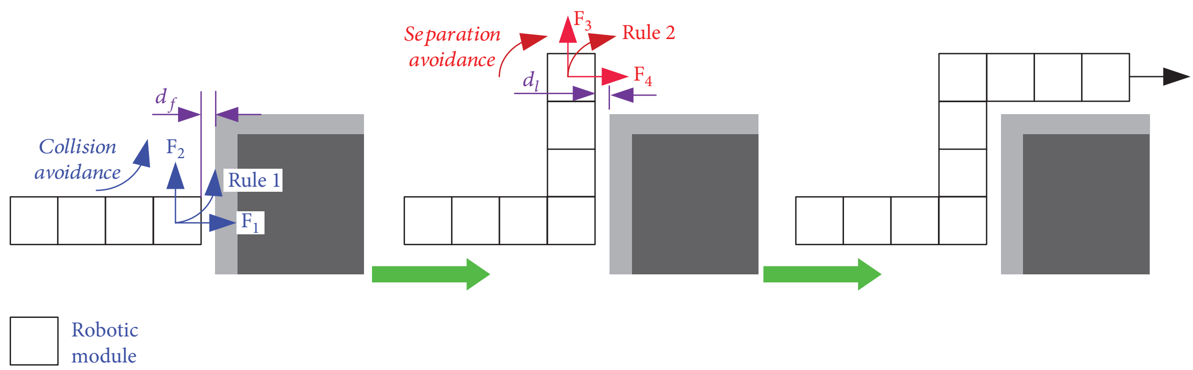

Figure 9: Environmental response mechanism.

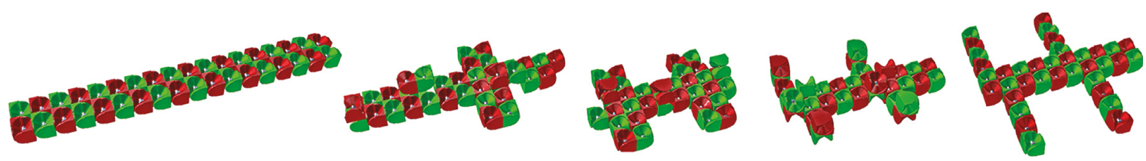

Figure 10: Self-reconfiguration process by the D0L-system in (2) without environmentally sensitive reconfiguration.

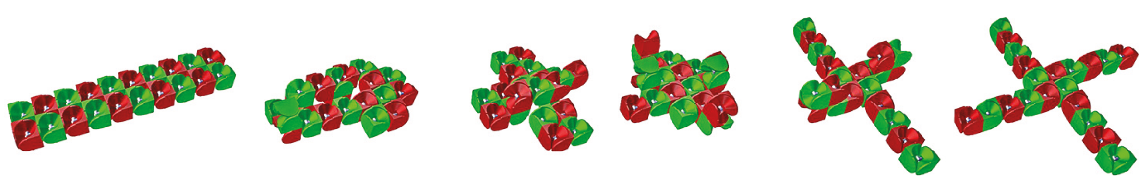

Figure 11: Self-reconfiguration process to a cross shape configuration.

system has grown out the desired structure in Figure 5, we say the self-reconfiguration process is convergent. Figure 11 shows another reconfiguration process to a cross-shape configuration.

The convergence of the proposed method by D0L-systems is illustrated in simulations. Both simulations in Figures 10 and 11 are repeated 50 times to illustrate the convergence of self-reconfiguration process by the proposed method. Simulation results show that all reconfigurations are convergent.

The proposed method is also scalable to module numbers by using the rewriting strategy of L-systems. Global configurations turn up to be self-similar by repeating substructures through interpreting rewriting rules. Several simulations are done on UBot robots with an increasing number of modules using the same L-system in (2). As shown in Figure 12, statistical results show that time steps for global convergence increase linearly as the module numbers. The linear relation verifies the scalability of the proposed method. Self-reconfiguration by the proposed method is no longer limited by the time-consuming global planning and the number of modules, which is the main constraint for optimal self-reconfiguration [4].

The convergence and scalability of the proposed method make full use of inspirations from plant growth. This introduces the variety of nature to the modular robot domain. While L-systems are opening a novel door for constructing target structure for self-reconfiguration, the natural pattern and construction morphology of plants provide a rich library for the design of global configurations for MSR robots.

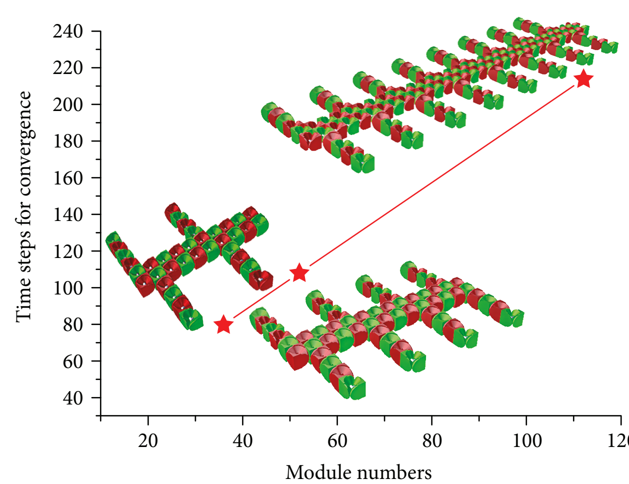

Figure 12: scalability to module numbers through rewriting.

4.2. Environmentally Sensitive Reconfiguration in Obstacle Space. Distributed self-reconfiguration by the proposed method is also sensitive to external environments by the use of parametric L-systems. Modules interact with neighboring environment independently by onboard sensors. The development of robotic structures depends on growth

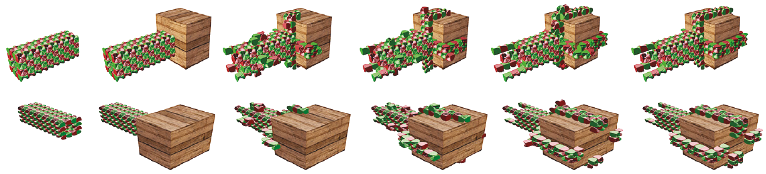

Figure 13: The touch response of robotic self-reconfiguration to an emergent box-shape object.

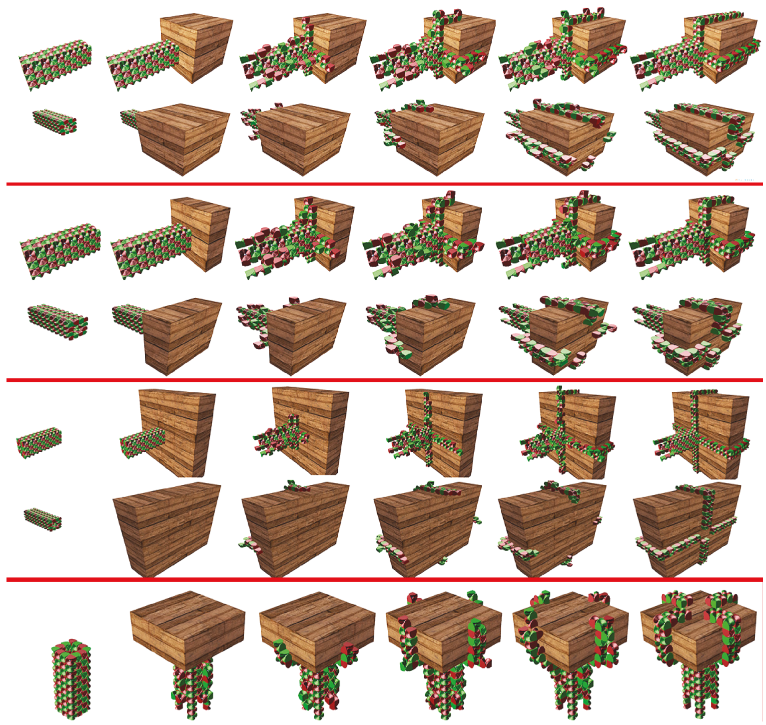

Figure 14: Environmentally sensitive reconfiguration of touching object in different scales.

planning of turtle modules. Turtle modules consider both internal and external conditions for local development planning according to parametric reproduction rules. And the following algorithm is implemented by turtle modules:

(i) Repeat

(ii) Step 1: collect internal and external information

(iii) Step 2: do motion planning according to environmentally sensitive rules

(iv) Until done

The touch-sensitive growth of plants can construct climbing or grasping structures. Using the reproduction rules in (4) and (5), robotic structures can grow along the surface of touching objects. Figure 13 shows that the UBot robot reconfigures to a multifinger-shaped structure by using the designed control method. Though no modules have geometric or position information of the touching object, local sense of independent modules can support the environmentally sensitive reconfiguration process. In this simulation, it is supported that new modules are generated continually at the bottom point for the global development.

This touch-sensitive self-reconfiguration is adaptive to geometric size of emergent objects. Because the development process relies on the local sensing and independent motion control of decentralized modules, the size of target objects has no geometry limit to single modules. As shown in Figure 14, reconfiguration results turn up to be self-adaption to emergent boxes. Though the result structure was not determinate previously, the interaction with environment and robotic development by turtle interpretation can guarantee variety of resultant configurations. This also indicates a potential adaption of self-reconfiguration to unknown environments.

5. Experimental Results

When the turtle module is doing a moving search work, it needs to have the sense of turtle coordinate system. The turtle coordinate translates along the development of robotic structure, from a turtle module to a new turtle module. As a result, even modules may be connected in various relative orientations, the turtle coordinate maintains an initially determined orientation, and branches develop along predefined global orientation.

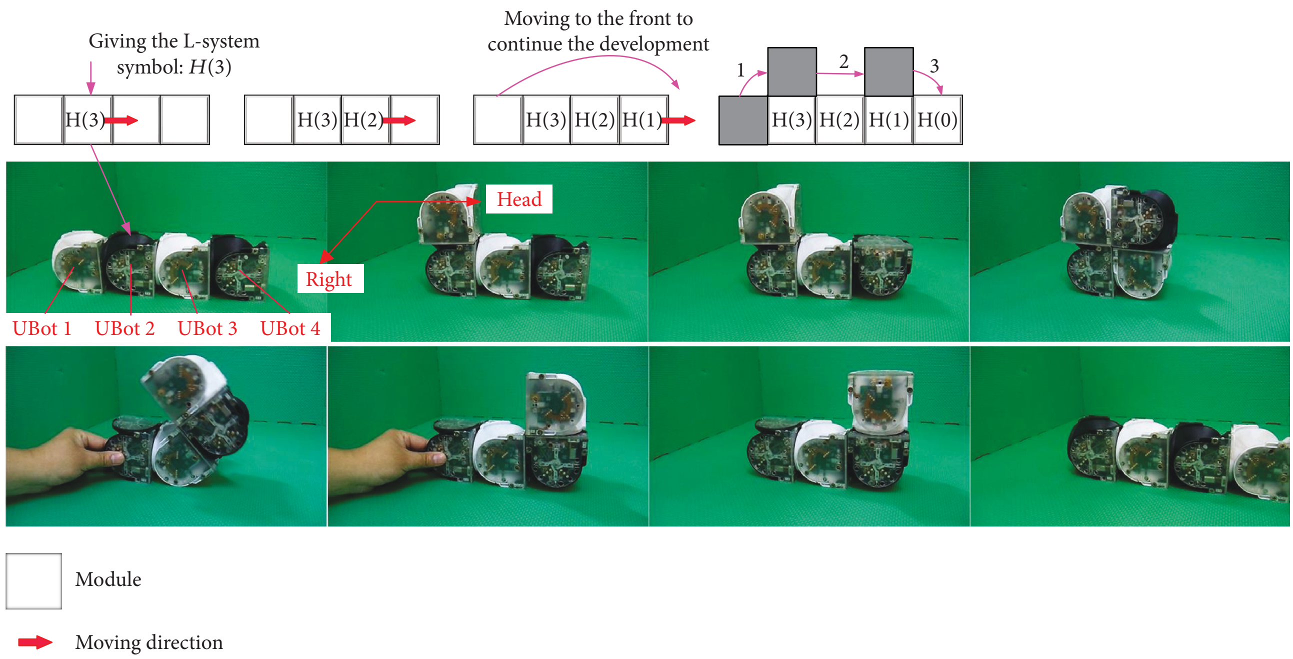

Figure 15: The hardware experiment for interpreting symbols $H(3)$.

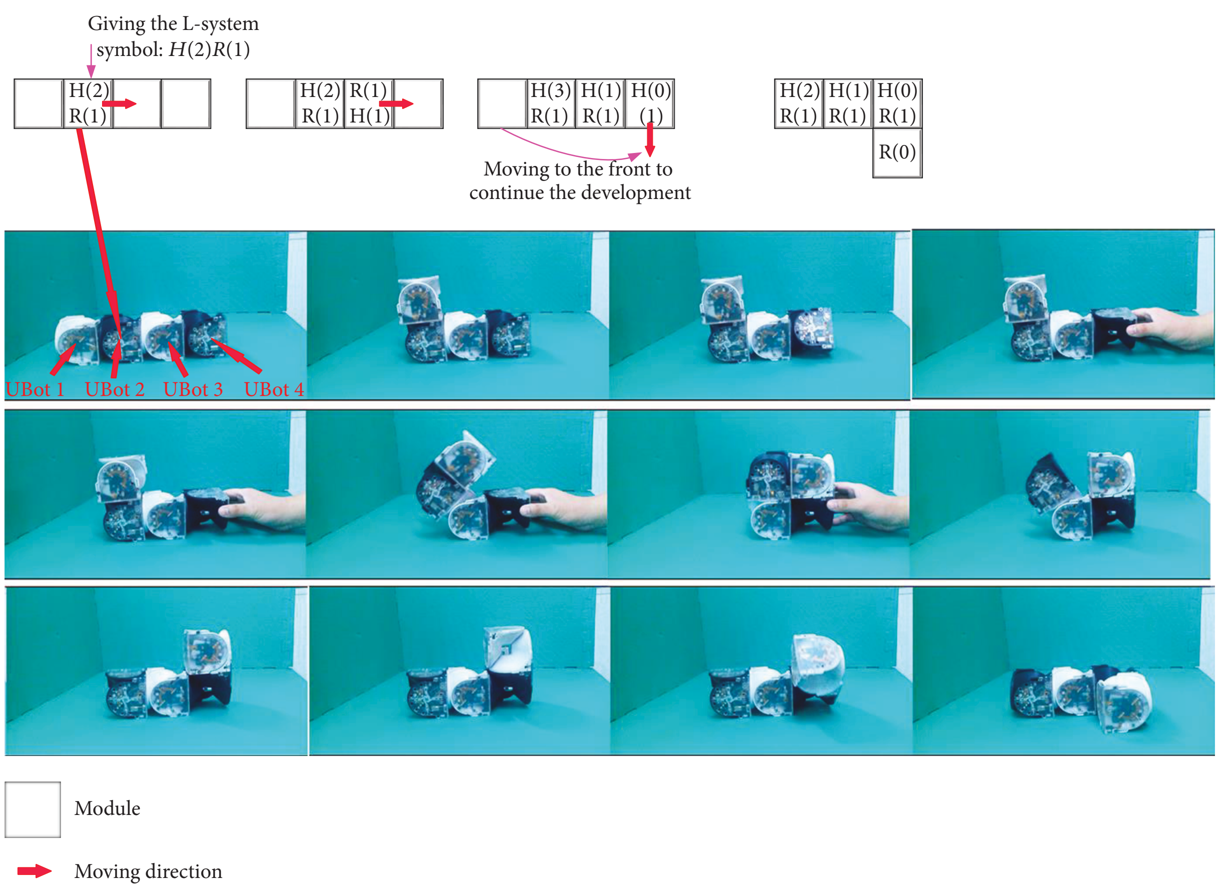

Figure 16: The hardware experiment for interpreting symbols $H(2)R(1)$.

Two simple experiments of a UBot robot are presented in Figures 15 and 16. The successful implementation of CA on UBot systems [24] provides reliable control for locomotion climbing gradient.

As shown in Figure 15, a UBot robot with four modules connected linearly. Self-reconfiguration starts when UBot2 receives L-system symbol $H(3)$. After translation through UBot3, UBot4 needs a module for symbol $H(0)$ in the heading direction. It then attracts UBot1 to the growing front. UBot1 needs assistance from modules on the moving path in order to get to needed position. Firstly, module UBot2 lifts UBot1 to the up lattice and then UBot3 lifts UBot4 in the same way. Secondly, UBot4 connects to UBot1 before UBot1 disconnects from UBot2. Lastly, UBot3 pushes UBot4 to the original position while lifting UBot1 to the developing front. The supplementary video (available here) records this self-reconfiguration experiment.

Both turtle modules and structure modules can provide assistance for the locomotion of mobile modules. As shown in Figure 15, movements of UBot1 contain assistance from modules UBot2 (structure module), UBot3 (structure module), and UBot4 (turtle module). Structure modules maintain the original position after assistance.

If the lattice at the turtle moving direction has no module, turtle modules will spare gradient [11, 41] to attract mobile modules in the system, for example, the process of UBot4 attracting UBot1 in Figure 15.

The gradient information spreads out through local communication in the decentralized system. All modules, including structure modules, turtle modules, and mobile modules, take part in the spread of gradient information. Mobile modules will climb gradient to the needed position while using CA managing local motion. Details about how CA manage movement of UBot modules along gradient can be found in our existing work [24].

The design of perpendicular joints makes flexible moving ability for UBot modules. As shown in Figure 16, UBot1 moves to another lattice neighboring to UBot4. The supplementary video records this self-reconfiguration experiment.

The transmission of symbol is kept in an absolute distributed way, which can keep the distributed nature of reconfiguration method. In a string type, L-system symbols are only translated from turtle modules to new attached mobile module, which then changes state to a turtle module and will handle the following interpretation. For example, UBot1 receives L-system symbols only when it gets to the needed position in both Figures 15 and 16. Turtle modules only interpret the first symbol in the string. Any symbol with zero length value ($s = 0$) will be deleted from the L-system string firstly before turtle modules interpret the following symbol.

Through the successful self-reconfiguration of those two experiments, we can see that the function of designed control strategy, including local translation of L-systems, extended symbol interpretation and surface locomotion by combining gradient attraction and CA. The results also proved that UBot modules can handle coordinate translation and relative localization in distributed manner. This extended interpretation of symbols achieves the degradation from global description to module-level predictions. Especially the degradation process contains no unpractical assumption for physical robots.

6. Conclusions

In this paper, we provide a novel contribution to this big target by considering how ideas from L-systems can solve the convergence of decentralized self-reconfiguration. The proposed method solved the convergent problem, which has been a corner stone on the way of bioinspired decentralized methods for self-reconfiguration of MSR robots. The self-reproduction of rewriting rules contributes to the scalability to module numbers. Reconfiguration times increase linearly to module numbers, and reconfiguration results show self-similarity by repeating substructures as described by L-system production rules.

More importantly, the self-reconfiguration process is sensitive to external environments. Parametric L-systems make it possible from local sensing to desired global phenomenon. Self-adaption results of touching objects in different size turn out to be similar in organization principles. This primary implementation indicates a decentralized mechanism for reconfiguration in real world.

The approach presented in this paper can be a starting point for further research from both theoretical and practical side, enriching each other with ideas coming from both MSR robots and L-systems. Such open research lines involve new developments in the application of the stablished theoretical framework of L-systems for the abstract representation of robots (and hence a deeper understanding of the theoretical possibilities) and also, from the practical side, the development of new abilities of physical robots inspired in the local relative position of the modules and the local encapsulation of the information. The study of techniques from both research areas can provide new solutions for matter of future research.

From the global structure point of view, the configuration of robotic systems keeps evolving over time. This is also the self-reconfiguration process. As the main aim of this article is introducing the implementation framework of L-systems for environmentally sensitive reconfiguration, the growing style of plants is translated to robotic development. That is why the structure modules in Figure 8 cannot change back to mobile modules. But the self-organizing process of MSR robots is dynamical over time in the modules' state. The structure modules in Figure 8 hold the ability to change to mobile modules for following reconfigurations over time. Robots can change to another reconfiguration for other targets at the current condition. The dynamical self-reconfiguration between different configurations from the current configuration and condition will be discussed in the future work.

Data Availability

The video data used to support the findings of this study are included within the supplementary files.

Conflicts of Interest

The authors declare no conflict of interest.

Authors' Contributions

Dongyang Bie, Iqbal Sajid, Jianda Han, Jie Zhao, and Yanhe Zhu contributed equally to this work.

Acknowledgments

The work reported in this paper is supported by the Joint Research Fund (U1713201) between the National Natural Science Foundation of China (NSFC) and Shenzhen. The work is also supported by the State Key Laboratory of Robotics (2018-O10).

Supplementary Materials

The video records simulations and experiments in this article. *(Supplementary Materials)*

References

[1] K. Gilpin and D. Rus, "Modular robot systems," *IEEE Robotics & Automation Magazine*, vol. 17, no. 3, pp. 38–55, 2010.

[2] S. S. R. Chennareddy, A. Agrawal, and A. Karuppiah, "Modular self-reconfigurable robotic systems: a survey on hardware architectures," *Journal of Robotics*, vol. 2017, Article ID 5013532, 19 pages, 2017.

[3] F. Hou and W. M. Shen, "On the complexity of optimal reconfiguration planning for modular reconfigurable robots," in *2010 IEEE International Conference on Robotics and Automation*, pp. 2791–2796, Anchorage, AK, USA, 2010.

[4] F. Hou and W. M. Shen, "Graph-based optimal reconfiguration planning for self-reconfigurable robots," *Robotics and Autonomous Systems*, vol. 62, no. 7, pp. 1047–1059, 2014.

[5] K. Stoy, "Lattice automata for control of self-reconfigurable robots," in *Robots and Lattice Automata*, pp. 33–45, Springer, 2015.

[6] M. Yim, W. M. Shen, B. Salemi et al., "Modular self-reconfigurable robot systems [grand challenges of robotics]," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 43–52, 2007.

[7] H. Ahmadzadeh, E. Masehian, and M. Asadpour, "Modular robotic systems: characteristics and applications," *Journal of Intelligent & Robotic Systems*, vol. 81, no. 3-4, pp. 317–357, 2016.

[8] H. Lakhlef, J. Bourgeois, H. Mabed, and S. C. Goldstein, "Energy-aware parallel self-reconfiguration for chains microrobot networks," *Journal of Parallel and Distributed Computing*, vol. 75, pp. 67–80, 2015.

[9] F. Hurtado, E. Molina, S. Ramaswami, and V. Sacristán, "Distributed reconfiguration of 2D lattice-based modular robotic systems," *Autonomous Robots*, vol. 38, no. 4, pp. 383–413, 2015.

[10] Z. Yang, Y. Wu, Z. Fu, J. Fei, and H. Zheng, "A unit-compressible modular robotic system and Its self-configuration strategy using meta-module," *Robotics and Computer-Integrated Manufacturing*, vol. 49, pp. 39–53, 2018.

[11] H. Bojinov, A. Casal, and T. Hogg, "Emergent structures in modular self-reconfigurable robots," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, pp. 1734–1741, San Francisco, CA, USA, 2000.

[12] A. Lyder, R. F. M. Garcia, and K. Stoy, "Mechanical design of odin, an extendable heterogeneous deformable modular robot," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 883–888, Nice, France, 2008.

[13] S. Mintchev, C. Stefanini, A. Girin et al., "An underwater reconfigurable robot with bioinspired electric sense," in *2012 IEEE International Conference on Robotics and Automation*, pp. 1149–1154, St. Paul, MN, USA, 2012.

[14] P. Zahadat, D. N. Hofstadler, and T. Schmickl, "Vascular morphogenesis controller: a generative model for developing morphology of artificial structures," in *Proceedings of the Genetic and Evolutionary Computation Conference on - GECCO '17*, pp. 163–170, Berlin, Germany, 2017.

[15] T. Schmickl, "How to engineer robotic organisms and swarms?," in *Bio-Inspired Self-Organizing Robotic Systems*, Y. Meng and Y. Jin, Eds., pp. 25–52, Springer, Berlin, Heidelberg, 2011.

[16] H. L. Ashe and J. Briscoe, "The interpretation of morphogen gradients," *Development*, vol. 133, no. 3, pp. 385–394, 2006.

[17] S. Kernbach, E. Meister, F. Schlachter et al., "Symbiotic robot organisms: REPLICATOR and SYMBRION projects," in *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems - PerMIS '08*, pp. 62–69, Gaithersburg, MD, USA, 2008, ACM.

[18] R. Thenius, M. Dauschan, T. Schmickl, and K. Crailsheim, "Regenerative abilities in modular robots using virtual embryogenesis," in *ICAIS 2011: Adaptive and Intelligent Systems*, A. Bouchachia, Ed., vol. 6943 of Lecture Notes in Computer Science, pp. 227–237, Springer, Berlin, Heidelberg, 2011.

[19] Y. Jin and Y. Meng, "Morphogenetic robotics: an emerging new field in developmental robotics," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 2, pp. 145–160, 2011.

[20] Y. Meng and Y. Jin, "Morphogenetic self-reconfiguration of modular robots," in *Bio-Inspired Self-Organizing Robotic Systems*, Y. Meng and Y. Jin, Eds., pp. 143–171, Springer, Berlin, Heidelberg, 2011.

[21] S. Murata and H. Kurokawa, "Self-organization of biological systems," in *Self-Organizing Robots*, pp. 19–35, Springer, Tokyo, Japan, 2012.

[22] J. Jones, S. Tsuda, and A. Adamatzky, "Towards *Physarum* robots," in *Bio-Inspired Self-Organizing Robotic Systems*, Y. Meng and Y. Jin, Eds., pp. 215–251, Springer, Berlin, Heidelberg, 2011.

[23] H. Ahmadzadeh and E. Masehian, "Modular robotic systems: methods and algorithms for abstraction, planning, control, and synchronization," *Artificial Intelligence*, vol. 223, pp. 27–64, 2015.

[24] Y. Zhu, D. Bie, S. Iqbal, X. Wang, Y. Gao, and J. Zhao, "A simplified approach to realize cellular automata for UBot modular self-reconfigurable robots," *Journal of Intelligent & Robotic Systems*, vol. 79, no. 1, pp. 37–54, 2015.

[25] K. Stoy, "Using cellular automata and gradients to control self-reconfiguration," *Robotics and Autonomous Systems*, vol. 54, no. 2, pp. 135–141, 2006.

[26] A. Vergara, Y. S. Lau, R. F. Mendoza-Garcia, and J. C. Zagal, "Soft modular robotic cubes: toward replicating morphogenetic movements of the embryo," *PLoS One*, vol. 12, no. 1, article e0169179, 2017.

[27] B. Salemi, M. Moll, and W.-M. Shen, "SUPERBOT: a deployable, multi-functional, and modular self-reconfigurable robotic system," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3636–3641, Beijing, China, 2006.

[28] Y. Zhu, D. Bie, X. Wang, Y. Zhang, H. Jin, and J. Zhao, "A distributed and parallel control mechanism for self-reconfiguration of modular robots using L-systems and cellular automata," *Journal of Parallel and Distributed Computing*, vol. 102, pp. 80–90, 2017.

[29] A. Lindenmayer, "Mathematical models for cellular interactions in development I. Filaments with one-sided inputs," *Journal of Theoretical Biology*, vol. 18, no. 3, pp. 280–299, 1968.

[30] A. Lindenmayer, "Developmental systems without cellular interactions, their languages and grammars," *Journal of Theoretical Biology*, vol. 30, no. 3, pp. 455–484, 1971.

[31] A. O. de Sá, N. Nedjah, and L. de Macedo Mourelle, "Distributed efficient localization in swarm robotics using min–max and particle swarm optimization," *Expert Systems with Applications*, vol. 50, pp. 55–65, 2016.

[32] X. Wang, H. Jin, Y. Zhu et al., "Serpenoid polygonal rolling for chain-type modular robots: a study of modeling, pattern switching and application," *Robotics and Computer-Integrated Manufacturing*, vol. 39, pp. 56–67, 2016.

[33] B. Runqiang, P. Chen, K. Burrage, J. Hanan, P. Room, and J. Belward, "Derivation of L-system models from measurements of biological branching structures using genetic algorithms," in *IEA/AIE 2002: Developments in Applied Artificial Intelligence*, T. Hendtlass and M. Ali, Eds., vol. 2358 of Lecture Notes in Computer Science, pp. 514–524, Springer, Berlin, Heidelberg, 2002.

[34] F. Aznar, M. Pujol, and R. Rizo, "L-system-driven self-assembly for swarm robotics," in *CAEPIA 2011: Advances in Artificial Intelligence*, J. A. Lozano, J. A. Gámez, and J. A. Moreno, Eds., vol. 7023 of Lecture Notes in Computer Science, pp. 303–312, Springer, Berlin, Heidelberg, 2011.

[35] H. Farooq, M. N. Zakaria, M. F. Hassan, and S. Sulaiman, "An approach to derive parametric L-system using genetic algorithm," in *IVIC 2009: Visual Informatics: Bridging Research and Practice*, H. Badioze Zaman, P. Robinson, M. Petrou, P. Olivier, H. Schröder, and T. K. Shih, Eds., vol. 5857 of Lecture Notes in Computer Science, pp. 455–466, Springer, Berlin, Heidelberg, 2009.

[36] P. Prusinkiewicz and A. Lindenmayer, *The Algorithmic Beauty of Plants*, Springer, 1990.

[37] M. Cieslak, A. N. Seleznyova, P. Prusinkiewicz, and J. Hanan, "Towards aspect-oriented functional-structural plant modelling," *Annals of Botany*, vol. 108, no. 6, pp. 1025–1041, 2011.

[38] P. Prusinkiewicz, "Graphical applications of L-systems," in *Proceedings on Graphics Interface '86/Vision Interface '86*, pp. 247–253, Vancouver, BC, Canada, 1986.

[39] D. Bie, G. Liu, Y. Zhang, J. Zhao, and Y. Zhu, "Modeling the fractal development of modular robots," *Advances in Mechanical Engineering*, vol. 9, no. 3, 2017.

[40] J. Knutzen, *Generating Climbing Plants Using L-Systems, [M.S. Thesis]*, Chalmers University of Technology, 2009.

[41] K. Stoy and R. Nagpal, "Self-reconfiguration using directed growth," in *Distributed Autonomous Robotic Systems 6*, pp. 3–12, Springer, 2007.