Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

Vision-Based Kinematic Configuration Recognition for Re-configurable Modular Robots

Junhao Li, Haifei Zhu*, Qi Zhang and Yisheng Guan
School of Electro-mechanical Engineering
Guangdong University of Technology
Guangzhou, Guangdong Province, China

Xiaoming Jiang
Guangdong Institute of Intelligent Manufacturing
Guangzhou, Guangdong Province, China

***Abstract*— Re-configurable modular robot systems are advantageous for their scalability and versatility. However, creating kinematic models for every new assembly is a time-consuming and error-prone task. In this paper, a vision-based kinematic configuration recognition method is proposed. The modular graph is firstly extracted by recognition of Quick Response (QR) Codes placed on each module. String information, obtained by searching kinematic chains on the modular graph, is then compared with that in a preset database, to figure out the current configuration. Experiments are conducted to verify the effectiveness of the proposed method, and to test the accuracy of the recognition. Results show that the intersection angle between the camera and the QR codes should be less than 45 degrees, to achieve the best recognition effect.**

I. Introduction

Modular robotic systems have been attracting a lot of attention, over the last three decades. Their features of versatility and reconfigurability bring along various benefits, such as good scalability and portability, as well as low cost. Most of the modular robotic systems can be divided into two types: Self- Reconfigurable Robots (SRRs) and Reconfigurable Modular Robots (RMRs) [5]. Compared to SRRs, single module of RMRs does not need some of the components that are necessary for self-reconfigurability, such as sensing, power supply, etc. Hence RMRs feature lower cost, smaller in size and simpler control systems. Usually, a centralized control system is required for each robot.

Modular robots with tree architecture pose more challenges to the reconfiguration algorithm. More specifically, configuration transformation involves chains, instead of single modules, causing more difficulties in the aspect of algorithm development. As increasingly more independent modules are installed on a robot, how these modules are included in the kinematic chain, and how the configuration of the robot will be reestablished from chains, are very challenging issues [2]. At present, most of RMRs without distributed independent controllers, must have their kinematic configuration manually preset before actual use, which is time-consuming and error-prone.

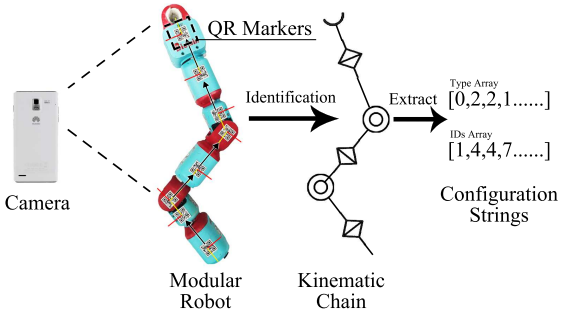


Fig. 1. A general process of the recognition.

To achieve the reconfiguration of chain-type RMRs without centralized off-line planners, Hou and Shen [2] proposed a reconfiguration planning algorithm for chain- type robots. A novel way called configuration string is provided, to represent the unlabeled configuration of robots, composed of homogeneous modules without IDs. A curtain configuration is represented by arrays that consisting of numbers of connectors at each node. A configuration can be represented as: [10 6 4 0][11 3 3 3][18 1 1 0], which is easy to use in software programs. Even though different systems for different types of robots are to be designed, configuration is represented in a similar way in all systems, to reduce software complexity and improve its scalability.

Lin *et al.* [1] also presented a vision-based scheme for kinematic model construction of tree-type RMRs. In this scheme, kinematic chains are constructed by geometric relationships between modules. A parent-searching method is used, to find all parent-child relations among detected modules, to avoid multiple results caused by tree-structure. According to this approach, the search process starts by selecting an end-effector as a child. Hence, arbitrary configurations without end-effectors cannot be identified in this

*Corresponding author (hfzhu@gdut.edu.cn). All authors are with School of Electromechanical Engineering, Guangdong University of Technology, Guangzhou, China, 510006.

This work is supported in part by the National Natural Science Foundation of China (Grant No. 51605096, 51705086),the Frontier and Key Technology Innovation Special Funds of Guangdong Province (Grant No. 2019A050503011), and the Program of Foshan Innovation Team of Science and Technology (Grant No. 2015IT100072).

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

scheme.

After analyzing assessing methods, in the prior art, to lower the barrier for non-experts and even inexperienced people, a new method for kinematic configuration recognition, along with relevant conceptions and definitions, are introduced. Next, the representation of the kinematic configuration of an RMR using graphs is described. Furthermore, the relationship between a kinematic configuration and its kinematic chains is discussed, as well as how to recognize an RMR configuration by its kinematic chains. A detailed method for the identification of kinematic chains will be mentioned later.

In Section II, some representations and the hardware platform are introduced. Section III describes a general framework of how to achieve recognition based on kinematic chains. Section IV introduces a detailed method of kinematic chain identification via QR Code. Some experiments and results are presented in Section V. Finally, conclusion and future work are discussed in Section VI.

II. Terminology and Representation

In this section, before defining the recognition framework, some of the representations, included in the proposed method, will be introduced.

A. Representation of Modules

RRMS (Reconfigurable Robot Modules and System) is a modular robotic system with a highly integrated and centralized control system. According to the relationship between the axes of joints and links, the modules of RRMS can be divided into joint modules and functional modules (Fig. 2). Joint modules include I-typed joints and T-typed joints. In I-typed joints, the revolute joint axis and link are collinear or parallel to each other. As for T-typed joints, the axes of joints and links are perpendicular to one another. Functional modules include various non-joint modules, such as control module, gripper module, and bridge module.

In joint modules, upper letters I and T indicate I-typed and T-typed joint modules. As for functional modules, letters G, B, C are used to represent the gripper, bridge and control modules respectively. When modules installed in an inverted way are detected, an I-typed module, for example, will be denoted as I'. A single quote will be added after the letter.

B. Representation of Configurations

A configuration is represented by a modular graph (Fig.3). Modular graphs can be used to represent both kinematic chains and configurations. Each node is a stand-alone module and each edge is the physical connection between modules. Every node has its unique global ID and a number called module type. Edges between nodes are directional, to avoid confusion caused by inversed installed modules. Due to the half-duplex serial communication used to control modules, nodes with the same ID are not allowed in the same configuration, to avoid communication confusion.

III. General Framework of Recognition

In order to recognize a robot's configuration, the first step is to identify the kinematic chains from its image. According to what has been introduced above, modular graphs are used to represent configurations and kinematic chains. A modular graph of the robot's configuration will be extracted and split into several individual subgraphs. Each graph represents a single sub kinematic chain of the whole configuration (Fig.4).

The modular graph of the configuration will be split into several subgraphs, following a tree structure, where all subgraphs have the same root node. First, the child-search process will start from a root node, which will be considered and recorded as a parent-node. For each parent-node, the algorithm will search any existing node in a specific area. Once a node is detected, it will be considered as a child-node and be recorded, while the child-node will become a new parent-node and the search process continues. If multiple nodes are detected within a parent-node's search area, this parent-node will be considered as a forking node and a new subgraph will be created as a new branch. The original graph will randomly select one of these nodes and continue extending, until no more nodes are detected in the latest parent-node search area. At this time, it becomes one of the subgraphs. The aforementioned subgraph will also continue extending from the forking node, but following a different route. This search process will not stop, until all possible search routes have been exhausted.

Once the search process is completed, subgraphs attained above will be transformed into arrays. In the following context, LG represents the length of a modular graph, with value equal to the number of nodes in a subgraph. In order to achieve the transformation, an array called Chain Array (CA)

(a) Control module (b) I-type joint module (c) T-type joint module

(d) Gripper module (e) Bridge module (f) Camera module

Fig. 2. The robotic modules.

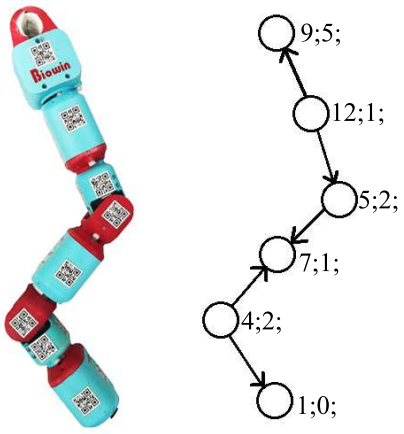


Fig. 3. A randomly built modular manipulator and its' modular graph. All joint modules in the kinematic chain are simplified into nodes. All edges are directional and all nodes are labeled with a unique ID and module type.

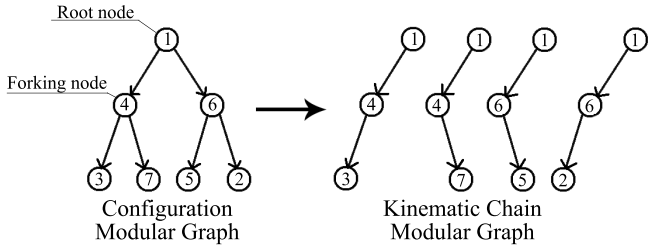


Fig. 4. A simple quadruped configuration modular graph on the left is splited into 4 subgraphs. Each of them represents a single kinematic chain. The number in nodes indicates their module ID, and the module type is omitted in this image.

with size L is defined. Next, all nodes in a subgraph will be sorted, according to the parent-child relationship, and added to the CA. Taking Fig.4 as an example, from left to right, the four subgraphs on the right will be transformed into [node1, node4, node3], [node1, node4, node7], [node1, node6, node5] and [node1, node6, node2] respectively. Further data extraction follows. Each module contains two integer data: module ID and module type. Modular type is represented by numbers, instead of letters, which were introduced in Section II, to simplify the algorithm. Table I shows some module types numberings and their corresponding modules. Both ID and module type will be extracted from CA and added to two new arrays, called ID Array (IDA) and Module Type Array (MTA).

Table II includes all data extracted from the quadruped configuration in Fig. 4. These data show that the robot has two front legs and two hind legs, performing only lateral movement. At the last step, these arrays are to be merged. Similarly, two new strings, called ID String (IDS) and Module Type String (MTS), are defined. ID array will be merged into ID string in an ascending order, while module type array follows its corresponding ID array. Hence the ID string of the configuration will be [1,4,3][1,4,7][1,6,2][1,6,5] and the module type string will be [0,3,2][0,3,2][0,3,2][0,3,2].

ID string and module type string represent the sequence of a configuration and the types of its modules. Therefore, an integral modular robot configuration can be reconstructed based on its ID string and module type string. Recognition can be achieved by comparing given ID string and module type string to preset ones in a database.

IV. Identification of Kinematic Chain

In order to extract the modular graph from an image, or split subgraphs from a configuration modular graph, a novel vision-based method is proposed.

Before the introduction of the detail steps, the content of QR codes and the definition of its direction are described. Each QR code contains a string in the format of ID; module type;. For example, 1;0; represents a control module with ID=1. As Fig.5 (a) demonstrates, each QR Code has 3 finder patterns that look like large squares with a smaller solid square within. These are located at the 3 corners of the QR Code, while there is no finder pattern at the lower right corner. This is considered as the normal orientation, as the front is facing upward.

More than one markers are placed on the surface of each module, such that at least one of the markers can be captured by the camera (Fig.5 (b)).

For I-typed modules, QR codes follow the direction of the rotation axis. For T-typed modules, they follow the direction of the swing arm.

The identification process of the whole configuration and the parent-child search method are based on a geometric

TABLE I
Representation of Modules

Module Type	Module Represented		Module Type	Module Represented
0	Control		3	Bridge
1	I-typed		5	Gripper
2	T-typed		6	Camera

(a) QR code (b) Joint modules with QR markers

Fig. 5. A example QR code and two modules attached multiple QR markers.

TABLE II

CONFIGURATION DATA EXTRACTED FROM FIG.4

CA	IDA	MTA
[node1,node4,node3]	[1,4,3]	[0,3,2]
[node1,node4,node7]	[1,4,7]	[0,3,2]
[node1,node6,node5]	[1,6,5]	[0,3,2]
[node1,node6,node2]	[1,6,2]	[0,3,2]

approach. All legal QR codes will be recognized as nodes. Each node has a specific area, called the recognition region (Fig.6).

In the case of I-typed modules and most of the functional

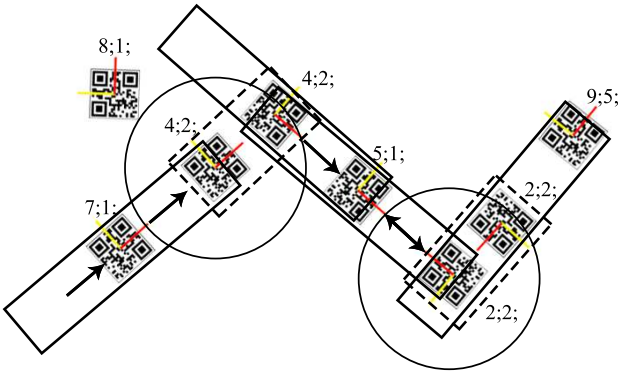


Fig. 6. A modular graph with QR codes and recognition region.

modules, the recognition region is a bidirectional rectangle (see node 7). In the case of bridge module, this region is a transversal one. The T-typed module has a special circle region, besides the rectangle one (see node 4 and 2). Hence two nodes in the same T-typed module have the same ID and module type.

Before the search algorithm starts, the whole image will be scanned to recognize all QR codes. The number of nodes (NN) will be calculated, and a node array sized NN will be defined. Node 7 is presumed to be the root node, in this example. The central point of node 4 is located in its recognition region. As the first detected node, in the T-typed module, its circular recognition region will be activated, instead of the rectangle one. Only nodes with the same ID will be detected in this circle. For the next detected node 4, the rectangle region will be reactivated. When an inverted installed module is detected, node 2, in this example, it will be assigned a module type, denoted as 2' in MTA. For isolated nodes, like node 8, its central point is out of any recognition region, so it will not be detected by any parent-node.

During the search process, as mentioned in Section III, once a node is detected and becomes a new parent-node, it will be instantly added to the CA. The length of the CA is

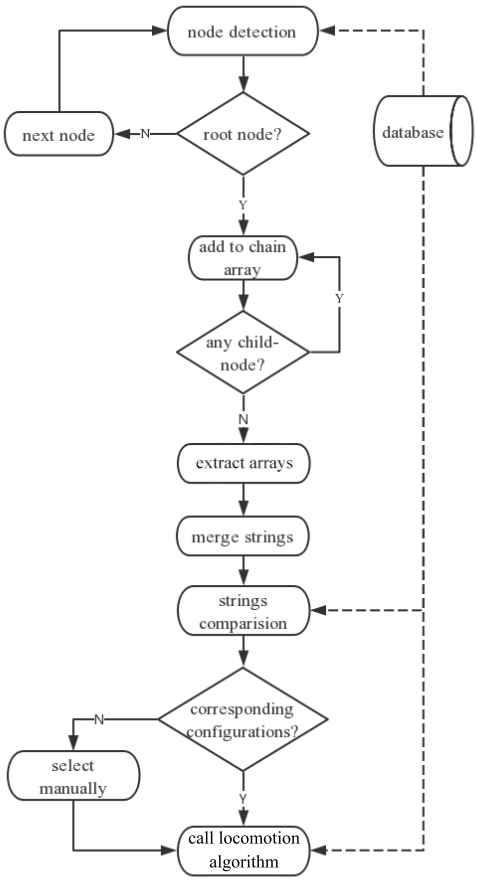


Fig. 7. Flowchart of the configuration recognition

set to be NN to avoid array overflow.

Following the search process, a regular expression is used to extract data from all CAs. Data of QR codes are stored in the form of string. The first number searched is considered as an integer and added to IDA. Similarly, the second number is added to MTA. Both IDA and MTA will merge into the respective string later for comparison purposes.

In order to reduce the complexity of the proposed recognition algorithm and thus have fewer difficulties, during implementation, angles between joints of the robots are ignored. Only sequence and module types of a configuration are considered. Locomotion algorithms for configurations are preset and stored in a database. Hence arbitrarily installed modular robots might not be possible to be recognized by this method. The corresponding subroutine will be called, when the configuration is recognized.

In the interest of portability, java was employed in all of the algorithms. Having the benefit of platform independence, multiple platforms are possible for the execution. One of the

Algorithm I Parent-child search function

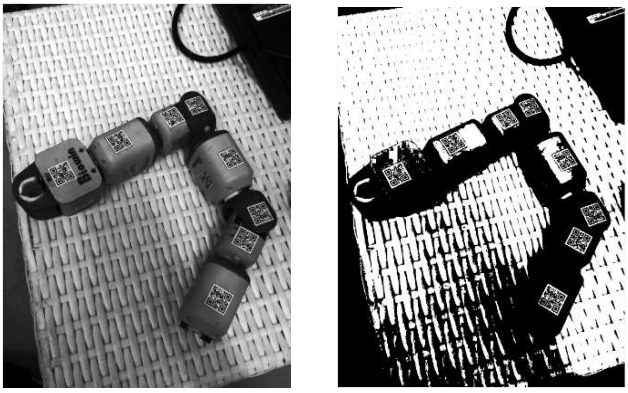
```
1:  define chainArray = ArrayList<node>;
2:  while ( searching ){
3:      set parentNode = latestNode in chainArray;
4:      if ( node is inRecogRegion of parentNode ){
5:          add node to chainArray;
6:      }
7:      else {
8:          stop searching
9:      }
10: }
```

presented applications is currently running on Android, for example.

As shown in Algorithm I, the search process keeps looping until no more nodes can be detected in the recognition region of the last node, in a CA. Next, an output is derived as an array of nodes, representing a kinematic chain. This process will not stop until all CAs in a modular graph reach their ends.

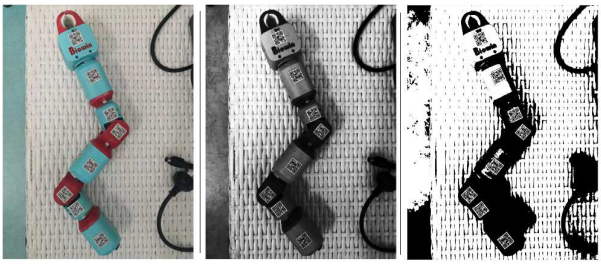
V. Experiments

In the presented method, OpenCV 3.0 is used, for image processing. Though a phone camera was used to take images, other RGB cameras will do just the same. Any RGB image input will be transformed into a greyscale image for further subsequent processing. Next, the image will be processed using a blur function to reduce noise. Then, it will be normalized and processed by the OpenCV equalizeHist function (Fig.8(a)). A threshold value will be set according to the average whiteness of the image, providing its binary version (Fig.8(b)). Once the image processing is completed, its output becomes the input to the identification process.

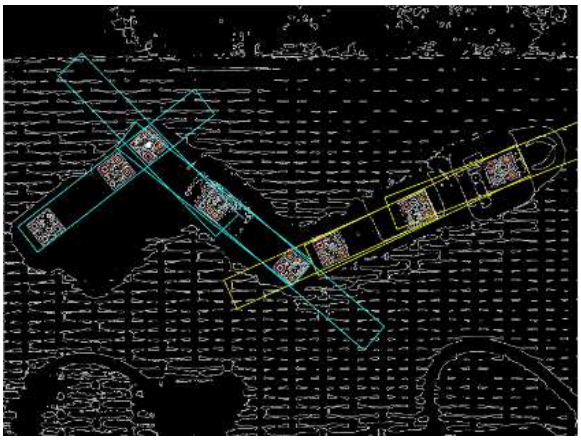


(a) Greyscale image (b) Binary image

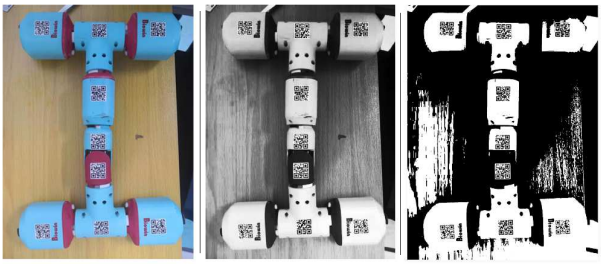
Fig. 8. Example images of a modular manipulator



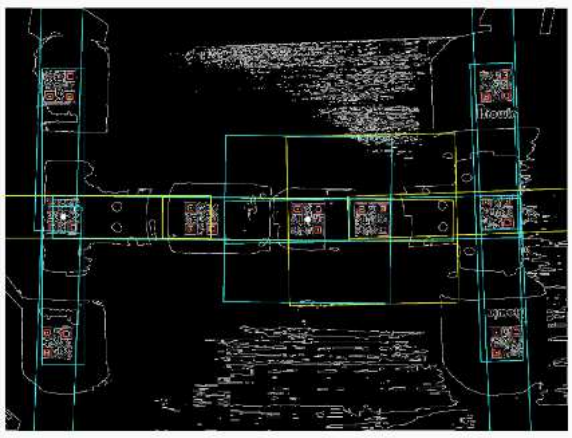
(a) Modular manipulator.



(b) Recognition image



(c) Four wheeled robot.



(d) Recognition image

Fig. 9. Two tested modular robots. The upper one is the same robot in Fig.8 but at a different configuration. The lower one is a four wheeled robot frame.

Fig. 10. The experiment is performed with an adjustable tripod, for quantitative adjustment. Distance between codes is set to 5cm.

Tests were carried out, two of which are illustrated in Fig. 9, where the proposed algorithm successfully identified the involved kinematic chains and provided the respective IDS and MTS.

The first experiment is shown in Fig. 9(a) and Fig.9(b), where an earlier version of QR code is used, hence the content of nodes cannot be recognized correctly. Therefore, the search process is interrupted at a detected T-typed module. The second experiment successfully identified all kinematic chains from the given image. IDs are simplified in this experiment, as modules of the same type use the same ID. Hence the derived result is of IDS=[1,2,2,12,8][1,2,2,12,8][1,12,5][1,12,5] and MTS=[0,2,2,3,1][0,2,2,3,1][0,3,1][0,3,1], proving that the proposed method can be applied to parallel kinematic chains.

To further test the recognition rate of the presented method, two quantitative experiments are performed. The following experiment is performed under ideal outdoor environmental illumination. The first experiment shown in Fig. 10 focuses on finding the maximum distance of the successful recognition. Due to the limited resolution of the input image, the QR code appears blurred, if the camera is positioned very far from the robot. In this experiment, the resolution of the input image is 1920x1080 and the scale of each code is set to 2cm, while the results are listed in Table III.

TABLE III
RESULTS OF EXPERIMENT I

Distance	Total nodes	Detected nodes	Recognition rate
30 cm	8	8	100%
40 cm	8	8	100%
50 cm	8	4	50%
60 cm	8	0	0%
70 cm	8	0	0%

The second experiment focuses on finding the maximum angle of the successful recognition. The angle of the camera keeps changing, while the distance remains the same. Results are listed in Table IV.

TABLE IV
RESULTS OF EXPERIMENT II

Angle	15°	30°	45°	60°	75°
Is Detected	true	true	true	true	false

VI. CONCLUSION

In this paper, a new configuration recognition method for modular robots is proposed. Graphs are used to represent modular robot configurations. An efficient algorithm, based on the high-level programming language Java, is developed to identify kinematic chains. All recognized kinematic chains are transformed into strings, for comparing to the preset database, to find the corresponding configuration. OpenCV 3.0 is used in this method to execute the image processing. The executed experiments show that recognition results are highly affected by ambient illumination, due to the uncertain threshold value in the binarization process.

Future work will focus on improving the accuracy of recognition, as well as the development of a recognition method for three-dimensional configurations, in order to achieve the identification of configurations of higher space complexity.

REFERENCES

[1] K. Lin, J. Rojas and Y. Guan, "A vision-based scheme for kinematic model construction of re-configurable modular robots," 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, 2017, pp. 2751-2757.

[2] Feili Hou and Wei-Min Shen, "Distributed, dynamic, and autonomous reconfiguration planning for chain-type self-reconfigurable robots," 2008 IEEE International Conference on Robotics and Automation, Pasadena, CA, 2008, pp. 3135-3140.

[3] M. Bordignon, K. Stoy and U. P. Schultz, "Generalized programming of modular robots through kinematic configurations," 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, 2011, pp. 3659-3666.

[4] C. Liu, M. Whitzer and M. Yim, "A Distributed Reconfiguration Planning Algorithm for Modular Robots," in IEEE Robotics and Automation Letters, vol. 4, no. 4, pp. 4231-4238, Oct. 2019.

[5] Y. Guan, L. Jiang, X. Zhangy, H. Zhang and X. Zhou, "Development of novel robots with modular methodology," 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, 2009, pp. 2385-2390.

[6] H. Ahmadzadeh, E. Masehian, "Modular robotic systems: Methods and algorithms for abstraction, planning, control, and synchronization,"Artificial Intelligence, vol. 223, pp. 27-64, 2005.

[7] M. Yim et al., "Modular Self-Reconfigurable Robot Systems [Grand Challenges of Robotics]," in IEEE Robotics and Automation Magazine, vol. 14, no. 1, pp. 43-52, March 2007.

[8] A. Spröwitz, R. Moeckel, M. Vespignani, S. Bonardi and A.J. Ijspeert, "Roombots: A hardware perspective on 3D self-reconfiguration and locomotion with a homogeneous modular robot,"Robotics and Autonomous Systems, vol. 62, no. 7, pp. 1016-1033, 2014.

[9] B. Salemi, M. Moll and W. Shen, "SUPERBOT: A Deployable, Multi-Functional, and Modular Self-Reconfigurable Robotic System," 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, 2006, pp. 3636-3641.

[10] T. Fukuda and S. Nakagawa, "Dynamically reconfigurable robotic system," Proceedings. 1988 IEEE International Conference on Robotics and Automation, Philadelphia, PA, USA, 1988, pp. 1581-1586 vol.3.