Configuration Identification for a Freeform Modular Self-reconfigurable Robot - FreeSN

Yuxiao Tu, *Student Member, IEEE,* and Tin Lun Lam, *Senior Member, IEEE*

***Abstract*—Modular self-reconfigurable robotic systems are potentially more robust and adaptive than conventional systems. This article proposes a novel freeform and truss-structured modular self-reconfigurable robot called FreeSN, containing node and strut modules. A node module contains a low-carbon steel spherical shell. A strut module contains two magnetic-based freeform connectors, which can connect to any position of the node module and provide spherical motions. Accurate configuration identification is essential for the automation of modular robot systems. This article presents a novel configuration identification system for FreeSN, including connection point magnetic localization, module identification, module orientation fusion, and system configuration fusion. A magnetic sensor array is integrated into the node module. A graph convolutional network-based magnetic localization algorithm is proposed, which can efficiently locate a variable number of magnet arrays under ferromagnetic material distortion. The module relative orientation is then estimated by fusing the magnetic localization result with the inertia moment unit and wheel odometry. Finally, the system configuration can be estimated, including the connection topology graph and the poses of modules. The configuration identification system is validated by a series of accuracy evaluation experiments and two library-based automation demonstrations based on closed-loop control.**

***Index Terms*—Cellular and modular robots, magnetic localization, self-reconfigurable, graph convolutional network (GCN), sensor fusion.**

Nomenclature

S_j	Strut module
V_j^u	Strut upper vehicle
V_j^n	Strut lower vehicle
N_i	Node module
H_i	Node horizontal plane
M_m	Magnetic sensor
T_k	Spherical triangle
P_{k*}	The *-th vertex of the spherical triangle T_k
${}^{*}_{**}\boldsymbol{R}$	Rotation matrix of body ** relative to body *
${}^{*}_{**}\hat{\boldsymbol{q}}$	Unit quaternion of body ** relative to body *
${}^{*}_{**}\boldsymbol{p}_{\#}$	Position of body # with respect to ** and expressed in the frame *
${}^{*}\hat{\boldsymbol{z}}_{**}$	Z-axis of body ** expressed in the frame *
${}^{*}\hat{\boldsymbol{r}}$	Vehicle target position relative to body *
${}^{*}_{**}\boldsymbol{\omega}_{\#}$	Angular velocity of body # with respect to ** and expressed in the frame * (rad/s)
$\boldsymbol{\omega}_{\#}$	Angular velocity of body # with respect to earth and expressed in the frame # (rad/s)
${}^{*}_{**}\omega_{V_j^{\#},f}$	Forward speed of the # vehicle of strut j with respect to ** and expressed in the frame * (rad/s)
${}^{*}_{**}\omega_{V_j^{\#},s}$	Steering speed of the # vehicle of strut j with respect to ** and expressed in the frame * (rad/s)
$\omega_{V_j^*,l}$	Rotation speed of the left wheel
$\omega_{V_j^*,r}$	Rotation speed of the right wheel
r_n	Radius of the node module
r_w	Strut wheel radius
l_w	Wheel interval of the vehicle
l_{S_j}	Central distance between connected node and strut S_j
η	Scalar part of the unit quaternion ${}^{H_i}_{S_j}\hat{\boldsymbol{q}}$
$\boldsymbol{\epsilon}$	Vector part of the unit quaternion ${}^{H_i}_{S_j}\hat{\boldsymbol{q}}$
$\boldsymbol{r}(i)$	i-th element of vector $\boldsymbol{r}$
$\otimes$	Hamilton product of quaternions
$\circ$	Hadamard product of two vectors or matrixs

I. Introduction

Modular self-reconfigurable robotic (MSRR) systems [1]–[5] are multi-robot systems consisting of a number of repeated robot modules, which can rearrange themselves into different configurations and adapt to new circumstances. Compared with the traditional fixed-morphology robots, MSRRs excel in unknown and unstructured environments and have strong self-adaptive and self-healing capabilities.

Over the past few decades, many MSRRs have been proposed based on different types of connectors, such as latch [6]–[8], hook [9], [10], electromagnet [11], permanent magnet [12]–[14], binder material [15], [16], and vacuum [17]. They have demonstrated the rich versatility and adaptability of MSRR in several aspects. However, many previous MSRR modules have a limited number of fixed location connectors, which makes large-scale MSRR systems vulnerable to misalignment between connectors due to tolerance stack [16]. The freeform MSRRs [1] have non-fixed point connectors and generally have high misalignment tolerance [4], [18] and a continuous and more flexible connection. Many early freeform MSRRs [19]–[21] are proposed, but they have weak connections and can only work in 2D. Several freeform MSRRs were proposed in recent years [16], [22]–[25], most of them can work in 3D, and they have demonstrated [25]–[27] that freeform MSRRs can better adapt to unstructured surfaces and unexpected changes of environment. The freeform connection brings many advantageous properties while posing challenges for position sensing.

This work was supported in part by the National Natural Science Foundation of China under Grant 62073274; in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2023B1515020089; and in part by the Shenzhen Institute of Artificial Intelligence and Robotics for Society under Grant AC01202101103. *(Corresponding author: Tin Lun Lam)*

Yuxiao Tu and Tin Lun Lam are with Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS), The Chinese University of Hong Kong, Shenzhen 518172, China. E-mail: yuxiaotu@link.cuhk.edu.cn, tllam@cuhk.edu.cn.

Many types of sensors have been applied for position sensing in previous MSRRs, such as potentiometers [6], [28]–[30] and optical or hall effect sensor encoders [9], [31], [32], but these sensors can only work for joints with one degree of freedom (DoF). Many of the existing higher-DoF spherical joints measure the joint orientation by several single-axis sensors [33], optical sensors [34], [35], and magnetic sensors [36]. However, most of them are not compact enough for tight space and weight constraints in modular robotic systems [2] or can hardly satisfy the freeform connection since they might require additional precisely aligned mechanical linkages.

In our previous work, we presented a freeform MSRR called FreeSN [24], combining the characteristics of freeform MSRR and truss-structured [1], [37]–[39] MSRR. FreeSN consists of strut modules and node modules. The node module is mainly a low-carbon steel spherical shell. The strut module contains two magnetic-based freeform connectors, which can connect and reconfigure to any position of the surface of the node module. FreeSN modules can be assembled and self-reconfigured into strut-node structural configurations, which can significantly improve the structural stability of the system and superimpose module capabilities by parallelizing the module motions. To realize automatic reconfiguration, the FreeSN system needs to accurately locate a variable number of connected struts through a $0.8\,\mathrm{mm}$ thick low-carbon steel spherical shell without external sensors. However, sensors can hardly be placed at the steel shell surface, limited by the free movement of the connector and the desired air gap between the connector and the spherical shell. The magnetometer cannot determine the orientation relative to the Earth since the magnetic field produced by the magnet array and the low-carbon steel spherical shell is far larger than the geomagnetic field. An optical system with cameras and fiducial markers is susceptible to interference from, for example, dust and light, especially in outdoor environments. The magnet array inside the connector naturally provides a strong and stable signal, which the steel shell can not block. A magnetic localization system has the potential to identify the FreeSN configuration accurately.

The permanent magnet localization method has been widely studied in the past decades. Many model-based localization algorithms model magnet behaves as a magnetic dipole, and various algorithms such as trilateration [40], nonlinear optimization [41]–[44], unscented Kalman filter (UKF) [45], and jacobian-based iterative method [46] were proposed to locate a known number of magnets. An optimization-based method was proposed [47] to locate a variable number of dipole sources by initializing a large number of dipole sources and optimizing their existence probabilities, but the maximum number of possible magnets needs to be indicated, and this influences the optimization efficiency. Due to the inaccuracy of the dipole at the surface of the magnet, some artificial neural network (ANN) based [48]–[51] and hybrid [52] algorithms were proposed to approximate the magnetic field and achieved good localization accuracy when tracking a single magnet. For more complex tasks, where the magnetic array is the tracking target or ferromagnetic materials such as steel exists, the magnetic field distribution is far from the traditional magnetic models, and inaccurate models can introduce significant localization errors. Convolutional neural network (CNN) was proven effective in dealing with nonlinearities introduced by ferromagnetic cores in electromagnetic modeling [53]. In our previous work [54], a graph convolutional network (GCN) based magnetic localization algorithm was preliminarily evaluated using FreeBOT [22] by classifying the non-uniform regions defined on the sphere. However, the localization accuracy is limited since the region division can hardly be extended, and the magnet position is not effectively encoded.

This article proposes a novel freeform MSRR called FreeSN. FreeSN evolves upon the work presented in [24] and [54]. Compared with the previous robot design [24], the node module integrates a magnetic sensor array, and a configuration identification system is proposed. The configuration identification system includes connection point magnetic localization, module identification, module orientation fusion, and system configuration fusion. The magnetic localization system includes the magnetic sensor array inside the node module and a GCN-based magnetic localization algorithm, which can locate all strut modules that connect to the node module. The feasibility of the magnetic localization system was preliminarily evaluated in [54]. A magnetic module identification approach is also proposed to identify the IDs of the adjacent modules. With the localization results from the proposed magnetic localization system, the relative orientation between the adjacent strut and node modules can be estimated by fusing with the data from the inertia measurement unit (IMU) and wheel odometry. The system configuration of a group of interconnected modules can also be estimated, including the pose of each module and the connection topology. With the estimated system configuration feedback, this article proposes a closed-loop controller for the differential driving vehicle of strut modules. The modules can perform system behaviors automatically by executing a sequence of control tasks from the library. Finally, we present the evaluation of the localization accuracies under the commercial optical motion capture system and demonstrate two system automation behaviors. The main contributions of the article are the following:

1) The mechanical design of a novel freeform MSRR called FreeSN, consists of strut and node modules, which can form truss structures and have continuous connections.
2) A connection point localization system for FreeSN that can locate a variable number of magnet arrays under the ferromagnetic environment using a GCN-based approach.
3) A configuration identification system for FreeSN that estimates the connection topology and the relative pose of each module.

The rest of this article is organized as follows. Section II introduces the robot design and mechanical performance of FreeSN. Section III proposes a magnetic localization and identification system that enables the node module to identify and track the connected strut modules. Section IV derives the orientation filter of modules and the system configuration fusion strategy. Section V introduces the vehicle controller design and the library-based system behavior. Section VI eval-

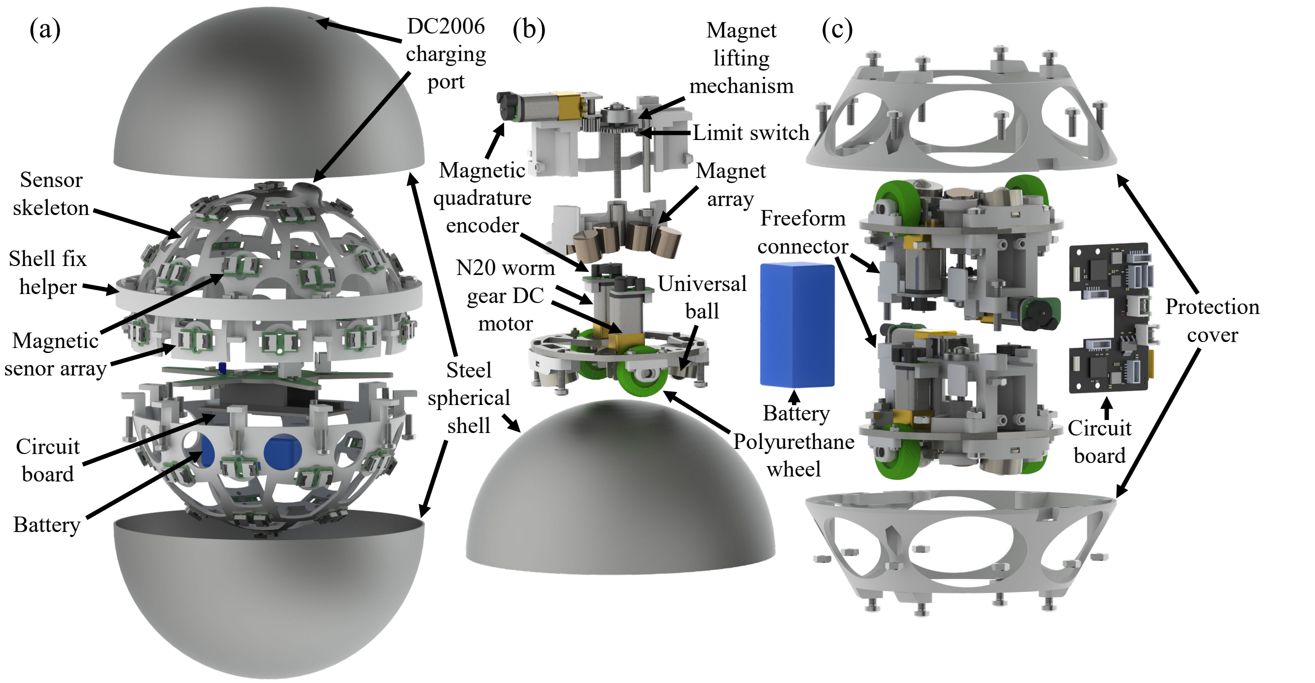

Fig. 1. The mechanical design of the FreeSN modules. (a) Node module. Exploded view. (b) Freeform connector. Exploded view. Partial section view. (c) Strut module. Exploded view.

uates the magnetic localization accuracy and the orientation filter accuracy. Section VII demonstrates the library-based behavior of the FreeSN system. Finally, Section VIII concludes this article.

II. Robot Design

The FreeSN system consists of node modules and strut modules. The surface of the node module is a low-carbon steel spherical shell. The strut module contains two freeform connectors, which can connect and reconfigure to any position of the node module.

This section first introduces the mechanical design and the integrated electronics of the current FreeSN. Compared with the initial version [24] of FreeSN, the node module is integrated with the magnetic sensor array and accompanying electronics, and the sensing units inside the strut module are introduced in detail. The specifications of the implementation in this article are then summarized.

A. Mechanical Design

As shown in Fig. 1(a), a node module contains two low-carbon steel hemisphere shells and integrated electronics. The upper hemisphere shell is fixed with the shell fix helper by epoxy. The 3D-printed sensor skeletons can be installed on the shell fix helper, and the lower shell is fixed with the upper shell. The magnetic sensor array is fixed on sensor skeletons by nylon rivets, and its arrangement is introduced in Section III-A. A hole with a diameter of $2\,\mathrm{mm}$ is drilled in the upper shell for charging. The effect of such a small hole on the magnetic attraction and robot motion is negligible.

The mechanical design of the freeform connector is shown in Fig. 1(b). It consists of a permanent magnet array, a magnet lifting mechanism, and a two-wheel differential driving vehicle. The magnet array produces a strong axial connection force, while the lifting mechanism makes the connection force variable, and the vehicle brings spherical freeform motion.

The magnetic attraction between the magnet array and the low-carbon steel spherical shell is mainly axial since the steel spherical shell is homogeneous, providing the possibility for the freeform connection. The magnets are arranged tangentially to the spherical surface of the node module.

The distance between the magnet array and the steel spherical shell is defined as the magnet height. A magnet lifting mechanism driven by a DC motor and a screw rod is used to change the magnet height and adjust the magnetic attraction force so that the connector can connect to or disconnect from the node module. When the magnet height approaches zero, the magnetic force reaches its maximum. The connector can disconnect from the node module by retracting the magnets until the magnetic attraction is small enough. The lifting mechanism can keep protruding the magnet array after the magnet height approaches zero, where the magnet array contacts the steel spherical shell and the wheels are not. The magnet height is regarded as negative in this state, and the strut length is variable. Although the actuation distance is small in this design, this motion still provides some powerful functions and can be easily improved in future designs if needed. For example, the magnet array can touch the external environment and generate a large thrust. The misalignment of connectors can be aligned by first protruding and then retracting the magnet array.

Since the magnetic attraction force is mainly axial, a two-wheel differential driving vehicle can provide spherical freeform motion and improve the connection strength. The vehicle consists of two DC motors with worm gear reducers, two polyurethane wheels, and two universal ball bearing casters. The magnetic attraction provides pressure on the polyurethane wheels. The friction on the two polyurethane wheels provides the spherical freeform motion driving force and the resistance to the shear force, bending moment, and torsion, which are vital to the connection performance.

As shown in Fig. 1(c), a strut module contains two symmetrically placed freeform connectors so that a strut module can connect to two node modules simultaneously. The protection cover protects the motors and electronics from external forces. Sufficient space is reserved for future extensions, such as environmental sensors.

B. Integrated Electronics

Since the FreeSN contains two different modules, the electronics are different.

1) Main Control Unit: The strut module uses dual Espressif ESP32-PICO-D4 microcontrollers as the main processor, which drives the six N20 DC motors by DRV8833 dual H-bridge motor driver chips. The microcontrollers also deal with the orientation filter and low-level control tasks.

A node module contains an ESP32-PICO-D4 microcontroller for magnetic array management and orientation filter tasks. An Allwinner H3 (Quad-core Cortex™-A7) controller supports the computing resources for magnetic localization and other higher-level tasks.

2) Sensing Unit: Each of the six N20 DC motors inside the strut module is equipped with a magnetic quadrature encoder. Two limit switches are installed for magnet height calibration. In this way, the magnet height can be controlled, and the wheel odometry of vehicles can be calculated. An MPU6050 6-axis inertia measurement unit (IMU) is also embedded.

A node module integrates an MPU6050 IMU and an array of 42 Melexis MLX90393 tri-axis magnetic sensors. The magnetic sensors are electrically connected by the flexible flat cable (FFC) and communicate with the Espressif ESP32-PICO-D4 microcontroller through the I2C bus.

3) Communication Unit: Both kinds of modules are capable of WiFi/Bluetooth communication. In this article, all modules and a centralized computer communicate through a WiFi router.

The strut module can transmit messages to the node module by changing the magnetic field. However, the transmission rate is limited by the motor speed. This type of communication is only used for magnetic module identification in this article.

C. Specification

The FreeSN contains a lot of parameters. Here, we present an implementation with the following specifications. The diameter of the spherical shell in the node module is $120\,\mathrm{mm}$, while the thickness is $0.8\,\mathrm{mm}$. The magnet array consists of eight N52 cylindrical magnets, with the middle four and the outer four of opposite polarity. The diameter of the magnets is $10\,\mathrm{mm}$ and the height is $10\,\mathrm{mm}$. Under such configurations, the minimum central angle between two connectors is less than $60°$, and a node module can be connected by 12 strut modules simultaneously. The robot modules can construct triangle substructures, which is vital for structural stability. The strut module and node module masses are $430\,\mathrm{g}$ and $310\,\mathrm{g}$. The robot specifications of the implementation are summarized in Table I.

TABLE I
SPECIFICATIONS OF FREESN

Specification	Value
Strut module	
- Magnet remanence	$1.47\,\mathrm{T}$ (N52)
- Magnet size	D: $10\,\mathrm{mm}$, H: $10\,\mathrm{mm}$
- Magnet number	8
- Magnet height range	$[-15\,\mathrm{mm}, 10\,\mathrm{mm}]$
- Driver motor rated speed (No load)	$20\,\mathrm{RPM}$ $(12\,\mathrm{V})$
- Lifting mechanism rated speed (No load)	$0.87\,\mathrm{mm\,s^{-1}}$ $(12\,\mathrm{V})$
- Polyurethane wheel diameter	$20\,\mathrm{mm}$
- Caster ball diameter	$8\,\mathrm{mm}$
- Module height	$77\,\mathrm{mm}$
- Module weight	$430\,\mathrm{g}$
Node module	
- External diameter	$120\,\mathrm{mm}$
- Spherical shell thickness	$0.8\,\mathrm{mm}$
- Magnetic sensor and node surface distance	$7\,\mathrm{mm}$
- Module weight	$310\,\mathrm{g}$

D. Connector Characterization

In this section, the magnetic performance and the connection strength are preliminarily evaluated. The connection strength can be characterized by normal strength, shear strength, bending strength, and torsional strength.

Fig. 2(a) shows the variations of the simulated magnetic attraction force in COMSOL Multiphysics and the measured normal strength with the distance between the magnet array and the steel spherical shell. The magnetic force is approximately $140\,\mathrm{N}$ at maximum when the magnet array is in contact with the shell and decreases as the distance increases. The magnetic force also decreases with the roll angle and pitch angle inclination, which are simulated in COMSOL Multiphysics and presented in Fig. 2(b). The magnetic force variations represent the alignment force that can be provided and, together with the demanded alignment force, determines the misalignment tolerance of the variables.

The normal strength equals the magnetic force when wheels are not in contact with the steel spherical shell, and the friction is small since the surface of the magnet array is smooth. When the wheels contact the shell, the magnetic force is distributed over the wheels, and the wheel friction increases while the normal strength decreases. The wheel friction approaches its maximum when the magnet array and shell are about to but not separated, while the normal strength decreases to about $79\,\mathrm{N}$. The shear, bending, and torsional strength reach their maximum at this state.

Under maximum wheel friction, we evaluated the shear strength and bending strength of the connector. The two strengths are different in different directions, caused by the distribution of the wheels. Five trials are performed in five different directions and on different bending moment arms,

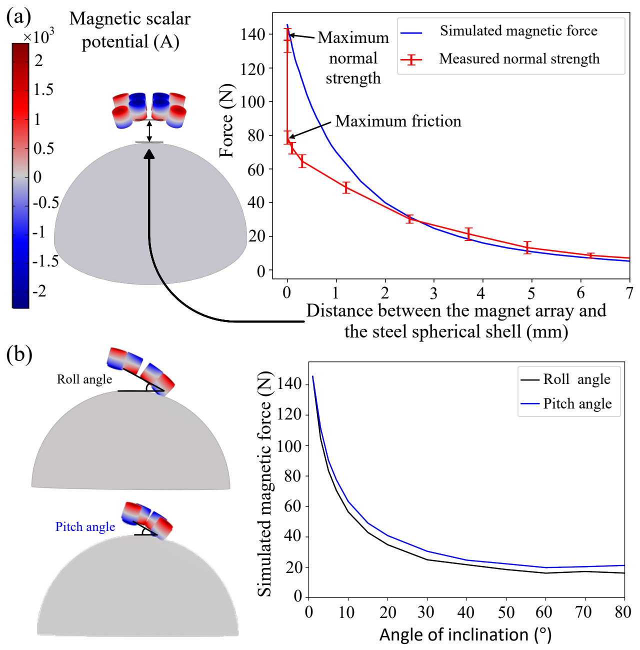

Fig. 2. (a) Variations of the simulated magnetic attraction force and the measured normal strength with the distance between the magnet array and the steel spherical shell. A simulation sample result is presented on the left side, which shows the magnetic scalar potential. (b) Variations of the simulated magnetic attraction force with the roll angle and pitch angle inclination.

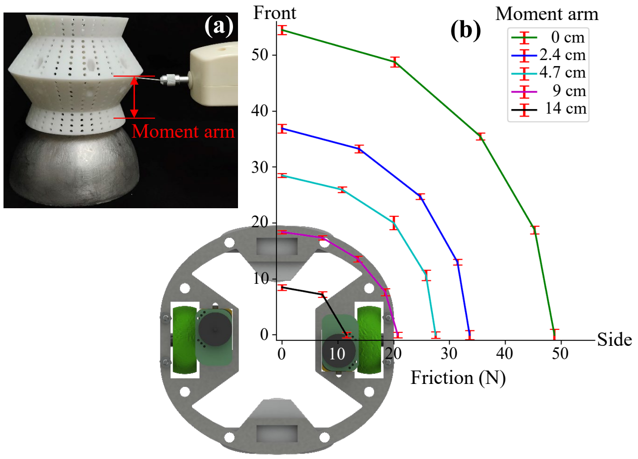

Fig. 3. Shear and bending strength test. (a) The forces are measured with a tensiometer and a 3D-printed helper. (b) The shear and bending strength test results in different moment arms and directions.

with the 3D-printed helper shown in Fig. 3(a). Under pure shear force, the maximum shear force is about $49\,\mathrm{N}$ in the side direction, as labeled in Fig. 3(b), while it is about $54\,\mathrm{N}$ in the front direction. The bending strength increases and then decreases with the moment arm, which also depends on the direction. The bending strength is relatively larger in the front direction when the moment arm is less than $4.7\,\mathrm{cm}$ and relatively larger in the side direction when the moment arm gets larger. The maximum bending moment is approximately $1.6\,\mathrm{Nm}$ in the front direction and $1.8\,\mathrm{Nm}$ in the side direction when the moment arm is about $9\,\mathrm{cm}$. The experimental results are concluded in Fig. 3(b). Assume that the modules form a strut-node-strut structure and connect to a fixed node module, the three modules can hold as a cantilever in the side direction, and the first two modules can hold as a cantilever in the front direction when the third module connects above the second module. The torsional strength is tested similarly. When only torsion is applied, the maximum torsion is about $1.3\,\mathrm{Nm}$.

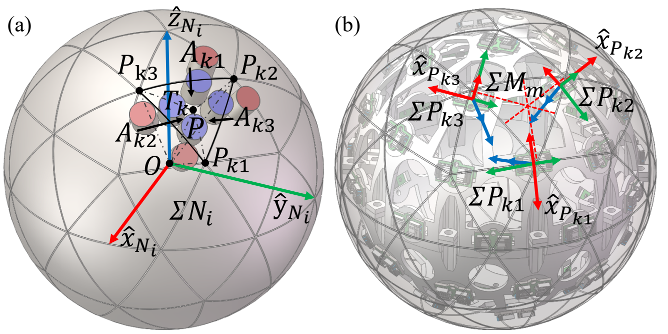

Fig. 4. (a) The spherical grid division and spherical triangle area notations. (b) The sensor array arrangement and frame definitions. The red dashed lines are the angle bisectors of the corresponding vertexes.

III. Magnetic Localization and Identification

The magnetic sensor array inside the node module can monitor the surrounding magnetic field and locate the nearby magnet arrays inside the freeform connector.

This section first introduces a spherical grid coordinate system, which guides the sensor arrangement and encodes the magnet array positions. Then, a GCN-based approach is proposed to locate a variable number of magnet arrays.

In addition, the magnet lifting mechanism can change the magnetic field by adjusting the magnet height. The strut module can send the encoded data by consecutively adjusting the magnet height, and node module can receive the data by decoding the nearest magnetic sensor data. The node module can identify the connected strut modules in this way, as introduced in Section III-C.

A. Spherical Grid System

The spherical surface can be divided into K spherical triangle areas based on the spherical area coordinate (SAC) system [55], which is singularity free and avoids planar mapping distortion. As shown in Fig. 4(b), the magnetic sensors are placed at the spherical projection of the vertices and edge midpoints of the regular icosahedron. The grid system is defined based on the sensor arrangement and produces 42 vertexes and 80 spherical triangle areas. The area division lines are shown in Fig. 4(a).

As shown in Fig. 4(a), the node frame ΣN_i is defined at its center O, and there exists a magnet array with the center point P. The magnetic localization algorithm in Section III-B aims to estimate the unit vector of the magnet array position relative to the frame ΣN_i, denoted as $\hat{\boldsymbol{p}} = \frac{\overrightarrow{OP}}{\left\|\overrightarrow{OP}\right\|}$. The SAC of $\hat{\boldsymbol{p}}$ is a better estimation target since it is relative to the grid system and avoids mapping distortion. The Cartesian coordinate and SAC can be interconverted with the grid system vertexes as follows.

As shown in Fig. 4(a), P lies inside a spherical triangle T_k with vertexes P_{k1}, P_{k2}, and P_{k3}. The Cartesian coordinates of the position vectors relative to the frame ΣN_i are denoted as $\hat{\boldsymbol{p}}_{k1} = \frac{\overrightarrow{OP_{k1}}}{\left\|\overrightarrow{OP_{k1}}\right\|}$, $\hat{\boldsymbol{p}}_{k2} = \frac{\overrightarrow{OP_{k2}}}{\left\|\overrightarrow{OP_{k2}}\right\|}$, and $\hat{\boldsymbol{p}}_{k3} = \frac{\overrightarrow{OP_{k3}}}{\left\|\overrightarrow{OP_{k3}}\right\|}$. The SAC of $\hat{\boldsymbol{p}}$ can be calculated as:

$$\boldsymbol{\mu}_k^T = \begin{bmatrix} \frac{A_{k1}}{A_k} & \frac{A_{k2}}{A_k} & \frac{A_{k3}}{A_k} \end{bmatrix} \tag{1}$$

where $A_k = \text{area}(\hat{\boldsymbol{p}}_{k1}, \hat{\boldsymbol{p}}_{k2}, \hat{\boldsymbol{p}}_{k3}) = A_{k1} + A_{k2} + A_{k3}$ is the area of the spherical triangle T_k. $A_{k1} = \text{area}(\hat{\boldsymbol{p}}, \hat{\boldsymbol{p}}_{k2}, \hat{\boldsymbol{p}}_{k3})$, $A_{k2} = \text{area}(\hat{\boldsymbol{p}}, \hat{\boldsymbol{p}}_{k3}, \hat{\boldsymbol{p}}_{k1})$, and $A_{k3} = \text{area}(\hat{\boldsymbol{p}}, \hat{\boldsymbol{p}}_{k1}, \hat{\boldsymbol{p}}_{k2})$ are the area of the sub-triangles. area is the spherical triangle area formula:

$$\begin{aligned} &\text{area}(\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2, \hat{\boldsymbol{p}}_3) = \\ &\qquad 2\tan^{-1}\left(\frac{\hat{\boldsymbol{p}}_1 \cdot (\hat{\boldsymbol{p}}_2 \times \hat{\boldsymbol{p}}_3)}{1 + \hat{\boldsymbol{p}}_1 \cdot \hat{\boldsymbol{p}}_2 + \hat{\boldsymbol{p}}_2 \cdot \hat{\boldsymbol{p}}_3 + \hat{\boldsymbol{p}}_3 \cdot \hat{\boldsymbol{p}}_1}\right) \end{aligned} \tag{2}$$

The SAC can also be converted to the Cartesian coordinate system by:

$$\hat{\boldsymbol{p}}^T = \boldsymbol{D}^{-1}\boldsymbol{b} \tag{3}$$

$$\boldsymbol{D} = \begin{bmatrix} (\hat{\boldsymbol{p}}_{k2} \times \hat{\boldsymbol{p}}_{k3})^T \\ (\hat{\boldsymbol{p}}_{k3} \times \hat{\boldsymbol{p}}_{k1})^T \\ (\hat{\boldsymbol{p}}_{k1} \times \hat{\boldsymbol{p}}_{k2})^T \end{bmatrix} - \tan\left(\frac{A_k}{2}\boldsymbol{\mu}_k \mathbf{1}^T \circ \begin{bmatrix} (\hat{\boldsymbol{p}}_{k2} + \hat{\boldsymbol{p}}_{k3})^T \\ (\hat{\boldsymbol{p}}_{k3} + \hat{\boldsymbol{p}}_{k1})^T \\ (\hat{\boldsymbol{p}}_{k1} + \hat{\boldsymbol{p}}_{k2})^T \end{bmatrix}\right) \tag{4}$$

$$\boldsymbol{b} = \tan\left(\frac{A_k}{2}\boldsymbol{\mu}_k \circ \begin{bmatrix} 1 + \hat{\boldsymbol{p}}_{k2} \cdot \hat{\boldsymbol{p}}_{k3} \\ 1 + \hat{\boldsymbol{p}}_{k3} \cdot \hat{\boldsymbol{p}}_{k1} \\ 1 + \hat{\boldsymbol{p}}_{k1} \cdot \hat{\boldsymbol{p}}_{k2} \end{bmatrix}\right) \tag{5}$$

B. Localization

Here, a GCN-based magnetic localization algorithm is introduced, which locates the center position of a variable number of magnet arrays.

As shown in Fig. 5(a), the input layer is the features of each triangle area, including the data from the magnetic sensors below three vertexes and the triangle geometric information. The magnetic sensor data is first normalized and then transformed to the area vertex frame ΣP. As shown in Fig. 4(b), the sensor frame ΣM_m is attached to the circuit board depending on the install direction, while the axis $\hat{\boldsymbol{z}}_{M_m}$ always points to the shell center O. The sensor frame can be transformed to the area vertex frames by rotating around its z-axis until its x-axis aligns with the angle bisector of the corresponding vertex. The geometric information is the one-hot encoding of the spherical triangle type, which is classified hierarchically based on geometric symmetricity and some manufacturing issues. In the proposed design, there are two base types.

The input layer features go through several GCN layers and multilayer perceptron (MLP) and finally output an 80x4 tensor of predictions, representing the existence probability C_k and SAC $\boldsymbol{\mu}_k$ of the magnet array center in each triangle area. As shown in Fig. 5(c), the SAC predictions are transformed to the Cartesian coordinate as candidate prediction by Eq. (3) if the corresponding existence probability prediction exceeds a threshold value. Then, the non-maximal suppression can effectively fix the possible multiple detections near the border of areas based on the assumption that the minimum central angle between two magnet arrays is always larger than a parameter. We set this parameter to 55° based on the mechanical parameters of FreeSN for robustness. Finally, we get the Cartesian coordinates ${}^{N_i}\hat{p}_j$ of the variable number of magnet arrays relative to the node frame ΣN_i.

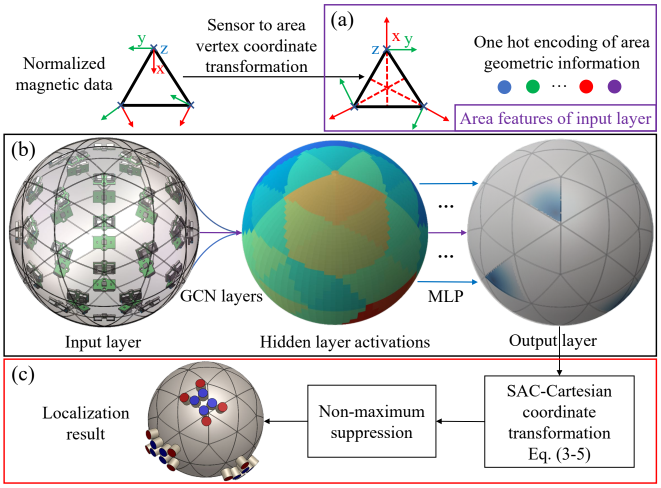
Normalized magnetic data
y
z
x
Sensor to area vertex coordinate transformation
(a)
x
y
z
One hot encoding of area geometric information
● ● ··· ● ●
Area features of input layer
(b)
GCN layers
···
···
MLP
Input layer
Hidden layer activations
Output layer
(c)
Localization result
Non-maximum suppression
SAC-Cartesian coordinate transformation Eq. (3-5)

Fig. 5. Magnetic Localization Algorithm. (a) The input features of the graph area, include the transformed magnetic data and the one-hot encoding of area geometric information. (b) The diagram of the graph convolutional network, visualizing the activations of a hidden layer and the output layer when using sample data. (c) After postprocessing, the algorithm outputs the centers of the connected magnet arrays.

The network can be trained using the following loss function:

$$\begin{aligned} \lambda_{coord}\sum_{k=0}^{K}\mathbb{1}_k^{mag}\|\boldsymbol{\mu}_k - \widehat{\boldsymbol{\mu}}_k\|^2 + \sum_{k=0}^{K}\mathbb{1}_k^{mag}(C_k - \widehat{C}_k)^2 \\ + \lambda_{nomag}\sum_{k=0}^{K}\mathbb{1}_k^{nomag}(C_k - \widehat{C}_k)^2 \end{aligned} \tag{6}$$

where λ_{coord} and λ_{nomag} are the weighted parameters, $\widehat{\boldsymbol{\mu}}_k$ and $\widehat{C}_k$ are the labels, and $\mathbb{1}_k^{mag}$ denotes if the center of a magnet array appears in the k-th triangular area.

Since the magnetic sensors are close to the magnets, and the distance between the magnet arrays is not too close, the model can be trained with a dataset where only one magnet array exists. This greatly reduces the difficulty of data collection.

The proposed network can be light-weighted. In the current implementation, the network contains only seven layers. The whole magnetic localization algorithm can run up to 20 Hz on the integrated microcontroller with a single core and up to 38 Hz with a quad-core.

C. Localization and Identification System

Adjacent module identification is a necessary function for an MSRR system. In FreeSN, the strut module can send data to the connected node module through low-frequency magnetic field communication, where the strut module changes

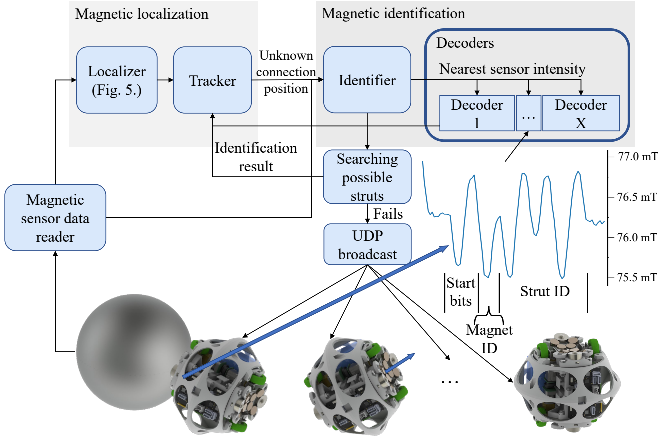

Fig. 6. The flowchart of the magnetic localization and identification system. The magnetic localization and identification systems work together to identify and track all connected strut modules.

the magnetic field by the magnet lifter and the node module monitors the magnetic field with the magnetic sensors. The communication speed might be slow, but it is reliable and does not require additional hardware. Magnetic identification is necessary only when two groups of modules are first connected. Otherwise, this procedure can be omitted since the system configuration is known.

As shown in Fig. 6, the magnetic localization and the identification systems work together to identify and track the connected strut modules. The magnetic localizer proposed in Section III-B reads the magnetic sensor data and predicts the position of the connected magnet arrays. Then the tracker tracks the magnet arrays by finding the closest reasonable prediction for each tracked magnet array. Once an unknown magnet array is tracked, the identifier first searches the possible strut modules from the interconnected module group. A centralized computer handles this task for now, as it may require the whole configuration of the module group. If the pose of some strut module satisfies the constraints of this unknown connection, we can think that this is the new tracked module, and we can know which side of this strut is connected.

If the search procedure fails, the identifier periodically broadcasts a user datagram protocol (UDP) package containing the estimated strut orientation information until successful identification. A magnetic decoder will be assigned to decode the magnetic field intensity of the nearest sensor based on the real-time magnet array position.

All the strut modules can receive the UDP package and judge which side of its magnet array can trigger this connection according to the received orientation information and its status. The possible magnet lifters will send the identification message through the magnetic field by consecutively retracting and protruding the magnet array. An example waveform of message ”01001101” received by the nearest magnetic sensor is shown in Fig. 6, where the first two bits are the start bits, the third bit is the magnet ID, and the rest bits are the strut ID. The magnetic field magnitude of the example waveform varies within $1.5\,\mathrm{mT}$, which is less than two percent of the magnetic field of the magnet array. The identification succeeds after the decoder decodes the message. Finally, the node module can track the strut module and establish the network connection for future functions.

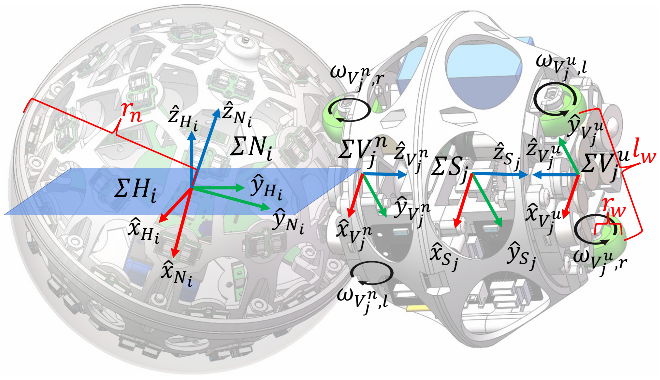

Fig. 7. Module coordinate system definition.

The low-frequency magnetic field communication data rate is limited by the speed of the lifter motor, depending on the input voltage and magnetic attraction load. In the presented design, the communication rate is approximately $2\,\mathrm{bits\,s^{-1}}$ when the input voltage is $12.6\,\mathrm{V}$, which is relatively slow but acceptable for the identification demands.

IV. System Configuration Fusion

With the module identification results, we can obtain the connection topology graph $\mathbb{G}$ for a group of interconnected FreeSN modules. The relative orientation between adjacent modules can be estimated by fusing the magnetic localization result, strut wheel odometry, and IMU. Finally, the system configuration can be estimated, including the relative pose of each module and $\mathbb{G}$.

In this section, the coordinate definition is first introduced. The module filters are then derived. Finally, a system configuration fusion strategy is proposed.

A. Coordinate Definition

As shown in Fig. 7, the node body frame ΣN_i locates at the sphere center, and we define a virtual horizontal plane H_i at the frame N_i origin. The z-axis $\hat{z}_{H_i}$ of the horizontal frame is always parallel with gravity, and the frame ΣH_i follows the rotation motion of the node module around the axis $\hat{\boldsymbol{z}}_{H_i}$ ideally.

As shown in Fig. 7, a strut module S_j connects to the node module, and the strut body frame ΣS_j is defined at the center of its body. The z-axis $\hat{\boldsymbol{z}}_{S_j}$ of frame ΣS_j is parallel with the magnet lifting direction, and $\hat{\boldsymbol{y}}_{S_j}$ is parallel with the line connecting the centers of the two polyurethane wheels at one side. The lower vehicle frame ΣV_j^n is parallel with the frame S_j, while the upper vehicle frame ΣV_j^u has the opposite y-axis and z-axis.

The wheel speeds are defined in Fig. 7. If a vehicle V_j^* moves on the node N_i without slipping, the forward speed ${}^{V_j^*}_{N_i}\omega_{V_j^*,f}$ and steering speed ${}^{V_j^*}_{N_i}\omega_{V_j^*,s}$ of the vehicle with respect to N_i and expressed in the frame ΣV_j^* can be estimated with the rotation speeds of its left wheel $\omega_{V_j^*,l}$ and right wheel

$\omega_{V_j^*,r}$, where $* \in \{u, n\}$ represents the upper vehicle or the lower vehicle.

$$\begin{bmatrix} {}^{V_j^*}_{N_i}\omega_{V_j^*,f} \\ {}^{V_j^*}_{N_i}\omega_{V_j^*,s} \end{bmatrix} = \begin{bmatrix} \frac{r_w}{\sqrt{4r_n^2 - l_w^2}} & \frac{r_w}{\sqrt{4r_n^2 - l_w^2}} \\ -\frac{r_w}{l_w} & \frac{r_w}{l_w} \end{bmatrix} \begin{bmatrix} \omega_{V_j^*,l} \\ \omega_{V_j^*,r} \end{bmatrix} \tag{7}$$

where r_w is the wheel radius, r_n is the radius of the node module, and l_w is the wheel interval of the vehicle.

Supposing that the strut module is perpendicular to the spherical tangent plane of the connected node module, the magnetic localization result ${}^{N_i}\hat{\boldsymbol{p}}_j$ points to the strut center. For the lower magnet array localization result, we have the estimated z-axis vector ${}^{N_i}\overline{\boldsymbol{z}}_{S_j} ={}^{N_i} \hat{\boldsymbol{p}}_j^n$, and ${}^{N_i}\overline{\boldsymbol{z}}_{S_j} = -{}^{N_i}\hat{\boldsymbol{p}}_j^u$ for the upper magnet array. The estimated z-axis can be represented in the horizontal frame by ${}^{H_i}\overline{\boldsymbol{z}}_{S_j} = {}^{H_i}_{N_i}\boldsymbol{R}{}^{N_i}\overline{\boldsymbol{z}}_{S_j}$.

B. Module Filters

The module filters include the node orientation filter, strut orientation filter, and yaw angle difference filter of adjacent nodes. The orientation filters follow the principle of the Madgwick filter [56] and estimate the module orientations relative to their reference frames. Since the node module is omnidirectionally symmetric, the node horizontal frames are chosen as the reference frames of the orientation filters. The yaw angle difference filter estimates the yaw angle difference between the horizontal frames of adjacent node modules.

1) Node Orientation Filter: The orientation filter of the node module follows the IMU implementation of the Madgwick filter [56], except that the third element of the measured angular rate expressed in its horizontal frame ${}^{H_i}_{E}\boldsymbol{\omega}^{gyro}_{N_i}(3)$ is set to zero before the integral, where frame ΣE is the earth frame. This filter estimates the orientation of frame ΣN_i relative to frame ΣH_i, denoted as the unit quaternion ${}^{H_i}_{N_i}\hat{\boldsymbol{q}}$ or the rotation matrix ${}^{H_i}_{N_i}\boldsymbol{R}$.

2) Strut Orientation Filter: The strut orientation filter estimates the orientation of the strut module relative to the horizontal frame of a connected reference node module, denoted as ${}^{H_i}_{S_j}\hat{\boldsymbol{q}}$. If only one connector of the strut S_j connects with a node module, the reference node is this module. If both connectors connect with a node module, we select the node module connecting the lower connector as the reference node. Without loss of generality, this reference node and the corresponding horizontal reference frame are denoted as N_i and ΣH_i.

The strut orientation filter estimates ${}^{H_i}_{S_j}\hat{\boldsymbol{q}}$ by fusing the data from wheel odometry, IMU, and magnetic localization result, following the framework of the Madgwick filter [56], which aligns the predefined reference direction with sensor data by optimization and integrates the angular velocity at each time step t, as:

$$\begin{cases} {}^{H_i}_{S_j}\boldsymbol{q}_t = {}^{H_i}_{S_j}\hat{\boldsymbol{q}}_{t-1} + \left(\frac{1}{2}\, {}^{H_i}_{S_j}\hat{\boldsymbol{q}}_{t-1} \otimes {}^{S_j}_{H_i}\overline{\boldsymbol{\varpi}}_{S_j} - \beta \frac{\nabla f_t}{\|\nabla f_t\|}\right)\Delta t \\ {}^{H_i}_{S_j}\hat{\boldsymbol{q}}_t = \frac{{}^{H_i}_{S_j}\boldsymbol{q}_t}{\left\|{}^{H_i}_{S_j}\boldsymbol{q}_t\right\|} \end{cases} \tag{8}$$

where $\frac{1}{2}\, {}^{H_i}_{S_j}\hat{\boldsymbol{q}}_{t-1} \otimes {}^{S_j}_{H_i}\overline{\boldsymbol{\varpi}}_{S_j}$ integrates the estimated angular velocity of S_j with respect to H_i expressed in the frame ΣS_j, β is the filter gain, and ∇f_t is the gradient of the objective function:

$$\begin{gathered} \min_{{}^{H_i}_{S_j}\hat{\boldsymbol{q}}} \frac{1}{2}\left\| f({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, \hat{\boldsymbol{a}}_{S_j}, {}^{H_i}\overline{\boldsymbol{z}}_{S_j})\right\|^2, \\ f({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, \hat{\boldsymbol{a}}_{S_j}, {}^{H_i}\overline{\boldsymbol{z}}_{S_j}) = \begin{bmatrix} f_g({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, \hat{\boldsymbol{a}}_{S_j}) \\ f_z({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, {}^{H_i}\overline{\boldsymbol{z}}_{S_j}) \end{bmatrix} \end{gathered} \tag{9}$$

$$f_g({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, \hat{\boldsymbol{a}}_{S_j}) = {}^{H_i}_{S_j}\hat{\boldsymbol{q}}^* \otimes {}^{H_i}\hat{\boldsymbol{g}} \otimes {}^{H_i}_{S_j}\hat{\boldsymbol{q}} - \hat{\boldsymbol{a}}_{S_j} \tag{10}$$

$$f_z({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, {}^{H_i}\overline{\boldsymbol{z}}_{S_j}) = {}^{H_i}_{S_j}\hat{\boldsymbol{q}} \otimes {}^{S_j}\hat{\boldsymbol{z}}_{S_j} \otimes {}^{H_i}_{S_j}\hat{\boldsymbol{q}}^* - {}^{H_i}\overline{\boldsymbol{z}}_{S_j} \tag{11}$$

where f_g aligns the theoretic gravity direction $\hat{\boldsymbol{g}}$ with the normalized accelerometer measurement $\hat{\boldsymbol{a}}_{S_j}$, f_z aligns the estimated z-axis $\overline{\boldsymbol{z}}_{S_j}$ of frame ΣS_j with the actual one $\hat{\boldsymbol{z}}_{S_j}$, and $\otimes$ represents the Hamilton product between quaternions. The gradient ∇f can be calculated as:

$$\nabla f = J_g({}^{H_i}_{S_j}\hat{\boldsymbol{q}})^T f_g({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, \hat{\boldsymbol{a}}_{S_j}) + J_z({}^{H_i}_{S_j}\hat{\boldsymbol{q}})^T f_z({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, {}^{H_i}\overline{\boldsymbol{z}}_{S_j}) \tag{12}$$

where J_g and J_z are the Jacobian matrices and the simplified gradient refers to Appendix A-A.

The angular velocity ${}^{S_j}_{H_i}\overline{\boldsymbol{\varpi}}_{S_j}$ of S_j with respect to H_i expressed in the frame ΣS_j can be estimated by fusing the angular velocities from the gyroscopes and the wheel odometry. The angular velocity of strut S_j and node N_i with respect to earth ΣE can be measured by their gyroscopes, denoted as $\boldsymbol{\omega}^{gyro}_{S_j}$ and $\boldsymbol{\omega}^{gyro}_{N_i}$. The wheel odometry data can be translated to the relative angular velocity of S_j with respect to frame N_i as:

$${}^{S_j}_{N_i}\boldsymbol{\omega}^{odom}_{S_j} = \begin{cases} \begin{bmatrix} 0 & {}^{V_j^n}_{N_i}\omega_{V_j^n,f} & {}^{V_j^n}_{N_i}\omega_{V_j^n,s} \end{bmatrix}^T \\ \begin{bmatrix} 0 & -{}^{V_j^u}_{N_i}\omega_{V_j^u,f} & -{}^{V_j^u}_{N_i}\omega_{V_j^u,s} \end{bmatrix}^T \end{cases} \tag{13}$$

depending on which side of the connector connects to the reference node.

However, the gyroscope suffers from a relatively large drift rate, and the wheel odometry is relatively inaccurate due to wheel slippage and systematic errors [57], and these errors can be eliminated with the gyrodometry [57]:

$${}^{S_j}_{N_i}\boldsymbol{\omega}^{go}_{S_j} = \begin{cases} {}^{S_j}_{N_i}\boldsymbol{\omega}^{odom}_{S_j}, \text{if } \left\|{}^{S_j}_{N_i}\boldsymbol{\omega}^{odom}_{S_j} - {}^{S_j}_{N_i}\boldsymbol{\omega}^{gyro}_{S_j}\right\| < \Omega_{th} \\ {}^{S_j}_{N_i}\boldsymbol{\omega}^{gyro}_{S_j} = \boldsymbol{\omega}^{gyro}_{S_j} - {}^{S_j}_{N_i}\boldsymbol{R}\, \boldsymbol{\omega}^{gyro}_{N_i}, \text{otherwise} \end{cases} \tag{14}$$

where Ω_{th} is the threshold parameter. Since the accelerometer can compensate for the drift of the gyroscope on the x-axis and y-axis relative to the earth frame, we can only apply the gyrodometry on the z-axis, and the relative angular velocity ${}^{S_j}_{H_i}\overline{\boldsymbol{\varpi}}_{S_j}$ can be estimated by combining the x-axis and y-axis of the gyroscope ${}^{H_i}_{E}\boldsymbol{\omega}^{gyro}_{S_j}$ and the z-axis of the gyrodometry ${}^{H_i}_{N_i}\boldsymbol{\omega}^{go}_{S_j}$ as:

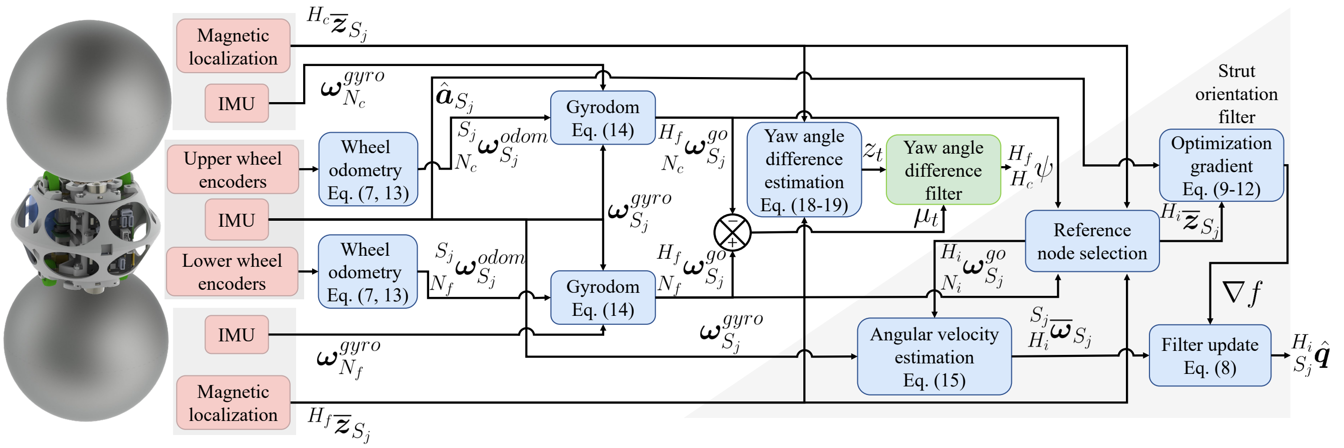
Magnetic localization
IMU
Upper wheel encoders
IMU
Lower wheel encoders
IMU
Magnetic localization
$^{H_c}\overline{\boldsymbol{z}}_{S_j}$
$\boldsymbol{\omega}^{gyro}_{N_c}$
$\hat{\boldsymbol{a}}_{S_j}$
Wheel odometry Eq. (7, 13)
$^{S_j}_{N_c}\boldsymbol{\omega}^{odom}_{S_j}$
Gyrodom Eq. (14)
$^{H_f}_{N_c}\boldsymbol{\omega}^{go}_{S_j}$
$\boldsymbol{\omega}^{gyro}_{S_j}$
Yaw angle difference estimation Eq. (18-19)
z_t
Yaw angle difference filter
μ_t
$^{H_f}_{H_c}\psi$
Strut orientation filter
Optimization gradient Eq. (9-12)
$^{H_i}\overline{\boldsymbol{z}}_{S_j}$
Reference node selection
Wheel odometry Eq. (7, 13)
$^{S_j}_{N_f}\boldsymbol{\omega}^{odom}_{S_j}$
Gyrodom Eq. (14)
$^{H_f}_{N_f}\boldsymbol{\omega}^{go}_{S_j}$
$^{H_i}_{N_i}\boldsymbol{\omega}^{go}_{S_j}$
∇f
$\boldsymbol{\omega}^{gyro}_{S_j}$
Angular velocity estimation Eq. (15)
$^{S_j}_{H_i}\overline{\boldsymbol{\omega}}_{S_j}$
Filter update Eq. (8)
$^{H_i}_{S_j}\hat{\boldsymbol{q}}$
$\boldsymbol{\omega}^{gyro}_{N_f}$
$^{H_f}\overline{\boldsymbol{z}}_{S_j}$

Fig. 8. The flowchart of the strut orientation filter and the yaw angle difference filter of adjacent nodes.

$$
{}^{S_j}_{H_i}\overline{\boldsymbol{\omega}}_{S_j} = \\
{}^{H_i}_{S_j}\boldsymbol{R}^T \left(\left({}^{H_i}_{S_j}\boldsymbol{R}\,\boldsymbol{\omega}^{gyro}_{S_j} \right) \circ \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \left({}^{H_i}_{S_j}\boldsymbol{R}\,{}^{S_j}_{N_i}\boldsymbol{\omega}^{go}_{S_j} \right) \circ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right) \tag{15}
$$

where $\circ$ is the Hadamard product and ${}^{H_i}_{S_j}\boldsymbol{R}$ is the rotation matrix corresponding to quaternion ${}^{H_i}_{S_j}\hat{\boldsymbol{q}}$. The detailed derivation refers to Appendix A-B.

3) Yaw Angle Difference Filter of Adjacent Nodes: When both connectors of a strut module connect to a node module respectively, the yaw angle difference ${}^{H_f}_{H_c}\psi$ between the two node horizontal frames can be estimated by the strut module S_j, where H_f is the parent node connecting the lower connector and H_c is the child node connecting the upper connector. The quaternion representing the relative orientation can be calculated as ${}^{H_f}_{H_c}\hat{\boldsymbol{q}} = \left[\cos\left({}^{H_f}_{H_c}\psi/2\right) \quad 0 \quad 0 \quad \sin\left({}^{H_f}_{H_c}\psi/2\right)\right]^T$, and ${}^{H_f}_{H_c}\psi$ also denoted as ψ_{S_j}. Here, a Kalman filter [58] based approach is applied, with the following state equation and observation equation:

$$
\begin{cases} \psi_t = \psi_{t-1} + u_t \Delta t + w_t \\ z_t = \psi_t + v_t \end{cases} \tag{16}
$$

where w_t and v_t are the noise variables, the control input u_t is the relative angular velocity, and z_t is the measurement of ψ. u_t can be calculated as the difference between the angular velocity of frame S_j relative to the frame H_f and frame H_c, as:

$$
u_t = {}^{H_f}_{H_f}\boldsymbol{\omega}^{go}_{S_j}(3) - {}^{H_f}_{H_c}\boldsymbol{\omega}^{go}_{S_j}(3) = {}^{H_f}_{N_f}\boldsymbol{\omega}^{go}_{S_j}(3) - {}^{H_f}_{N_c}\boldsymbol{\omega}^{go}_{S_j}(3) \tag{17}
$$

The measurement z_t can be estimated by aligning the projections ${}^{H_c}\hat{\boldsymbol{z}}^{proj}_{S_j}, {}^{H_f}\hat{\boldsymbol{z}}^{proj}_{S_j}$ of the magnetic localization results of two nodes on the horizontal frame as:

$$
z_t = \operatorname{arctan2}\left(({}^{H_c}\hat{\boldsymbol{z}}^{proj}_{S_j} \times {}^{H_f}\hat{\boldsymbol{z}}^{proj}_{S_j})(3), {}^{H_c}\hat{\boldsymbol{z}}^{proj}_{S_j} \cdot {}^{H_f}\hat{\boldsymbol{z}}^{proj}_{S_j} \right) \tag{18}
$$

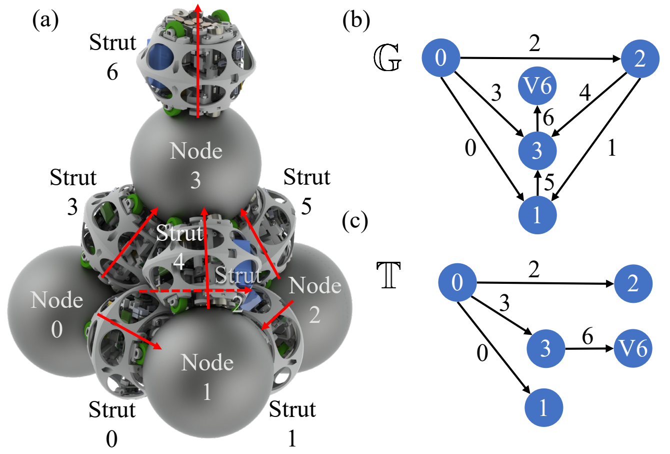

Fig. 9. (a) An example FreeSN configuration, where the arrow side of the red line is the upper connector. (b) The graph representation of the system topology. (c) A minimum spanning tree of the graph, which is used to estimate the system configuration.

$$
\begin{cases} {}^{H_i}\boldsymbol{z}^{proj}_{S_j} = {}^{H_i}\overline{\boldsymbol{z}}_{S_j} \circ \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}^T \\ {}^{H_i}\hat{\boldsymbol{z}}^{proj}_{S_j} = \dfrac{{}^{H_i}\boldsymbol{z}^{proj}_{S_j}}{\left\| {}^{H_i}\boldsymbol{z}^{proj}_{S_j} \right\|}, \quad i = f, c \end{cases} \tag{19}
$$

The measurement update steps are applied only when the norms of projections $\left\|{}^{H_f}\boldsymbol{z}^{proj}_{S_j}\right\|, \left\|{}^{H_c}\boldsymbol{z}^{proj}_{S_j}\right\|$ are larger than a threshold value, where the estimated measurement z_t is reliable.

The flowchart of the strut orientation filter and the yaw angle difference filter is summarized in Fig. 8.

C. System Configuration Fusion

For each group of inter-connected modules, the connection relationship can be represented as a directed graph, denoted as $\mathbb{G}$. The directed edges in the graph represent the strut modules, while the nodes represent the node modules. We define that the direction of the edge is the same as the z-axis of the strut frame. If a connector of some strut module is not connected to a node module, we add a virtual node to the graph. The graph can be automatically generated with the module identification results. An example configuration containing seven strut modules and four node modules is shown in Fig. 9(a), and Fig. 9(b) shows the corresponding graph, where the node V6 is a virtual node, and the virtual node ID is defined the same as the strut ID.

The system configuration can be solved as a tree structure. As shown in Fig. 9(c), we can maintain a minimum spanning tree $\mathbb{T}$ with the root node N_r. The pose of all modules relative to the root horizontal frame ΣH_r can be solved with Alg. 1, where edge outputs the edge in the graph that connects the given nodes, DFS traverses the tree via depth-first search, l_{S_j} is the center distance between the connected node and strut S_j, and $\boldsymbol{p}$ is the module position.

Algorithm 1 Tree-based system configuration fusion

```
for $N_n, N_m$ in $\mathrm{DFS}(\mathbb{T})$ do
    if $N_n$ is the parent node in $\mathbb{G}$ then
        $S_j = \mathrm{edge}(\mathbb{G}, N_n, N_m)$
        ${}^{H_r}_{H_m}\hat{\boldsymbol{q}} = {}^{H_r}_{H_n}\hat{\boldsymbol{q}} \otimes {}^{H_n}_{H_m}\hat{\boldsymbol{q}}$
        ${}^{H_r}_{S_j}\hat{\boldsymbol{q}} = {}^{H_r}_{H_n}\hat{\boldsymbol{q}} \otimes {}^{H_n}_{S_j}\hat{\boldsymbol{q}}$
        $\hat{\boldsymbol{d}} = {}^{H_r}\hat{\boldsymbol{z}}_{S_j}$
    else
        $S_j = \mathrm{edge}(\mathbb{G}, N_m, N_n)$
        ${}^{H_r}_{H_m}\hat{\boldsymbol{q}} = {}^{H_r}_{H_n}\hat{\boldsymbol{q}} \otimes {}^{H_m}_{H_n}\hat{\boldsymbol{q}}^*$
        if $N_m$ is virtual then
            ${}^{H_r}_{S_j}\hat{\boldsymbol{q}} = {}^{H_r}_{H_n}\hat{\boldsymbol{q}} \otimes {}^{H_n}_{S_j}\hat{\boldsymbol{q}}$
        else
            ${}^{H_r}_{S_j}\hat{\boldsymbol{q}} = {}^{H_r}_{H_m}\hat{\boldsymbol{q}} \otimes {}^{H_m}_{S_j}\hat{\boldsymbol{q}}$
        end if
        $\hat{\boldsymbol{d}} = -{}^{H_r}\hat{\boldsymbol{z}}_{S_j}$
    end if
    ${}^{H_r}_{N_m}\hat{\boldsymbol{q}} = {}^{H_r}_{H_m}\hat{\boldsymbol{q}} \otimes {}^{H_m}_{N_m}\hat{\boldsymbol{q}}$
    ${}^{H_r}\boldsymbol{p}_{N_m} = {}^{H_r}\boldsymbol{p}_{N_n} + 2l_{S_j}\hat{\boldsymbol{d}}$
    ${}^{H_r}\boldsymbol{p}_{S_j} = {}^{H_r}\boldsymbol{p}_{N_n} + l_{S_j}\hat{\boldsymbol{d}}$
end for
```

The system configuration can be further optimized when $\mathbb{G}$ contains cycles, which is common in a truss-structured MSRR system. One approach is to solve the following optimization problem:

$$\begin{aligned} \min_{\tilde{\psi}_{S_j}, {}^{H_i}\tilde{\boldsymbol{q}}_{S_j}, S_j \in \mathbb{G}} \sum_{c \in \mathrm{circ}(\mathbb{G})} & \left(\lambda_1 \left\| \sum_{S_j \in c} \mathrm{dir}(S_j) \tilde{\psi}_{S_j} \right\|^2 + \right. \\ & \left. \lambda_2 \left\| \sum_{S_j \in c} \mathrm{dir}(S_j)\, {}^{H_r}\tilde{\boldsymbol{z}}_{S_j} \right\|^2 \right) \\ + \sum_{S_j \in \mathbb{G}} & \left(\lambda_3 \left\| {}^{H_r}\tilde{\boldsymbol{z}}_{S_j} - {}^{H_r}\hat{\boldsymbol{z}}_{S_j} \right\|^2 + \lambda_4 \left\| \tilde{\psi}_{S_j} - \psi_{S_j} \right\|^2 \right) \\ \text{s.t.} \quad & \left\| {}^{H_i}\tilde{\boldsymbol{q}}_{S_j} \right\| = 1 \end{aligned} \tag{20}$$

where $\tilde{\psi}_{S_j}$ and ${}^{H_i}\tilde{\boldsymbol{q}}_{S_j}$ are the optimization variables. $\mathrm{circ}(\mathbb{G})$ finds the fundamental set of cycles of the graph [59]. $\mathrm{dir}(S_j)$ outputs 1 if the direction of strut S_j in $\mathbb{G}$ is the same as the cycle traversal direction and outputs -1 otherwise. $\lambda_1 \ldots \lambda_4$ are the weighting parameters.

The configuration optimization combines all filter information in the cycles. It can reduce the accumulative error in long chain structures and provides more accurate system configuration, which is useful for tasks that rely on the whole system configuration.

V. System Automation

With the feedback from the configuration identification system, we design the closed-loop controller for the differential driving vehicle. The modules can show system behaviors by executing a sequence of control tasks. In this section, a strut position controller is first introduced, which reaches the target orientation by following the shortest arc. Then, a library-based automation system is briefly introduced, demonstrating the module behaviors in Section VII.

A. Position Control

Assuming that a vehicle V_j^* connects to the node N_i, the vehicle can reach the desired target position ${}^{H_i}\hat{\boldsymbol{r}}$ with the orientation filter feedback. H_i can be replaced with N_i so that the control is relative to the frame ΣN_i.

Intuitively, the shortest path to the target position is the circular arc connecting the current vehicle position ${}^{H_i}\hat{\boldsymbol{z}}_{V_j^*}$ and the target position ${}^{H_i}\hat{\boldsymbol{r}}$. The target direction vector can be projected to the vehicle frame ΣV_j^* as

$${}^{V_j^*}\boldsymbol{r}^r = {}^{H_i}_{V_j^*}\boldsymbol{R}^T \left({}^{H_i}\hat{\boldsymbol{r}} - {}^{H_i}\hat{\boldsymbol{z}}_{V_j^*} \right) \tag{21}$$

where ${}^{H_i}_{V_j^*}\boldsymbol{R}$ is the rotation matrix calculated from the corresponding unit quaternion ${}^{H_i}_{V_j^*}\hat{\boldsymbol{q}}$ and can be calculated as

$$\begin{cases} {}^{H_i}_{V_j^n}\boldsymbol{R} = {}^{H_i}_{V_j^u}\boldsymbol{R} \circ \begin{bmatrix} \mathbf{1} & -\mathbf{1} & -\mathbf{1} \end{bmatrix} = {}^{H_i}_{S_j}\boldsymbol{R} \\ {}^{H_i}\hat{\boldsymbol{z}}_{V_j^n} = -{}^{H_i}\hat{\boldsymbol{z}}_{V_j^u} = {}^{H_i}\ \hat{\boldsymbol{z}}_{S_j} \end{cases} \tag{22}$$

There are two parameters to describe the arc, i.e., the arc central angle θ_j^* and the arc azimuth ϕ_j^*, which are calculated as

$$\begin{cases} \theta_j^* = \arccos\left({}^{V_j^*}\boldsymbol{r}^r(3) + 1 \right) \\ \phi_j^* = \mathrm{arctan2}\left({}^{V_j^*}\boldsymbol{r}^r(2), {}^{V_j^*}\boldsymbol{r}^r(1) \right) \end{cases} \tag{23}$$

Since the vehicle is implemented as differential driving, we adopt the first steering to the desired direction and then moving forward strategy. The desired forward velocity can simply be proportional to the central angle of the arc if the vehicle is facing the desired direction, and the desired steering velocity is proportional to the arc azimuth unless the vehicle is very close to the target direction, as:

$${}^{V_j^*}_{N_i}\omega_{V_j^*,f} = \begin{cases} K_f \theta_j^*, & \text{if } |\phi_j^*| < \Phi_{th} \\ 0, & \text{otherwise} \end{cases} \tag{24}$$

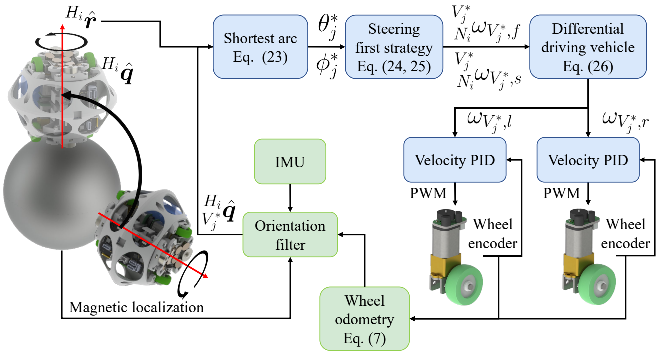
${}^{H_i}\hat{\boldsymbol{r}}$
${}^{H_i}\hat{\boldsymbol{q}}$
Shortest arc Eq. (23)
θ_j^*
ϕ_j^*
Steering first strategy Eq. (24, 25)
${}^{V_j^*}_{N_i}\omega_{V_j^*,f}$
${}^{V_j^*}_{N_i}\omega_{V_j^*,s}$
Differential driving vehicle Eq. (26)
$\omega_{V_j^*,l}$
$\omega_{V_j^*,r}$
IMU
Velocity PID
Velocity PID
PWM
PWM
${}^{H_i}_{V_j^*}\hat{\boldsymbol{q}}$
Orientation filter
Wheel encoder
Wheel encoder
Wheel odometry Eq. (7)
Magnetic localization

Fig. 10. Block diagram of the shortest-arc strut orientation controller.

$$
{}^{V_j^*}_{N_i}\omega_{V_j^*,s} = \begin{cases} K_s\phi_j^*, & \text{if } \theta_j^* > \Theta_{th} \\ 0, & \text{otherwise} \end{cases} \tag{25}
$$

where K_f and K_s are the control gains,and Φ_{th} and Θ_{th} are the threshold values.

Having obtained the forward and steering velocities, the wheel speeds can be solved from the following equation:

$$
\begin{bmatrix} \omega_{V_j^*,l} \\ \omega_{V_j^*,r} \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{4r_n^2 - l_w^2}}{2r_w} & -\frac{l_w}{2r_w} \\ \frac{\sqrt{4r_n^2 - l_w^2}}{2r_w} & \frac{l_w}{2r_w} \end{bmatrix} \begin{bmatrix} {}^{V_j^*}_{N_i}\omega_{V_j^*,f} \\ {}^{V_j^*}_{N_i}\omega_{V_j^*,s} \end{bmatrix} \tag{26}
$$

The wheel speeds are then controlled by the PID controllers with feedback from the wheel encoders. The vehicle can reach the desired position in this way. The vehicle can also reach the desired orientation ${}^{H_i}\hat{\boldsymbol{q}}$ by first reaching the corresponding position and then steering to the desired orientation. A whole flowchart of the above controller is presented in Fig. 10.

B. Library-based Behavior

We design a lot of parameterized meta-tasks, currently including the control tasks and synchronization tasks. The user can design the behavior file by indicating the system configurations and a sequence of meta-tasks in a graphic interface. The library contains lots of robot behaviors in different configurations.

With the real-time feedback from the configuration fusion system in Section IV, the automation system searches the candidate behaviors by matching the system topology with the library. After a behavior is chosen by the user, the modules defined in the behavior file and the actual modules are automatically mapped, including the module IDs and the struts directions. Then, the specific meta-tasks can be automatically generated in real time, while the commands of the low-level control tasks will be sent to the robot modules.

A centralized computer estimates the global configuration, assigns the control task target to the modules, and synchronizes the tasks while the modules share their configurations with adjacent modules and finish the control tasks using motors distributively. A robot behavior can be executed in this procedure, and a brief diagram of the library-based behavior automation is shown in Fig. 11.

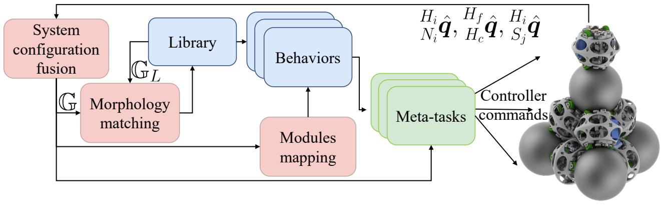
System configuration fusion
Library
Behaviors
$\mathbb{G}_L$
$\mathbb{G}$
Morphology matching
Modules mapping
Meta-tasks
${}^{H_i}_{N_i}\hat{\boldsymbol{q}}$, ${}^{H_f}_{H_c}\hat{\boldsymbol{q}}$, ${}^{H_i}_{S_j}\hat{\boldsymbol{q}}$
Controller commands

Fig. 11. Library-based system automation.

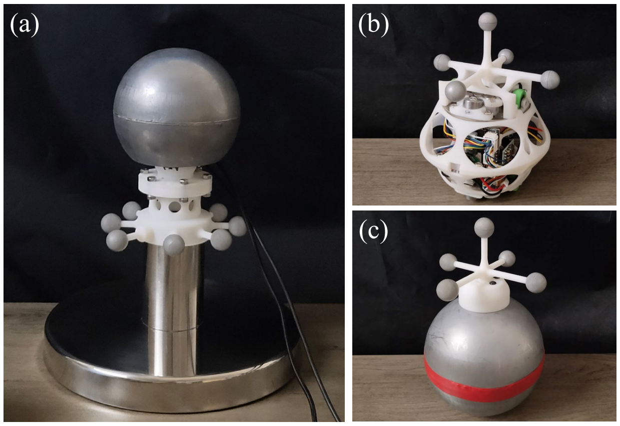

Fig. 12. Experimental setup. (a) A node module fixed on a counterweight base. (b) The strut module with optical markers add-on. (c) The node module with optical markers add-on.

VI. Localization Accuracy Evaluation

In this section, we evaluate the accuracy of magnetic localization and module filters in three experimental setups. The data collection process for network training and accuracy evaluation is first introduced.

A. Data Collection Setup

The module poses are recorded using the OptiTrack motion capture system with eight Prime 13 cameras on a truss of $2\,\mathrm{m}$, $2\,\mathrm{m}$, and $1.6\,\mathrm{m}$ in length, width, and height, respectively. The mean ray error is $0.328\,\mathrm{mm}$ during calibration, which decides the measurement uncertainty. For easier data collection, we design a particular node module in which the sensor skeleton is fixed on a counterweight base. As shown in Fig. 12(a), eight optical markers are fixed on the base so that the base orientation is monitored. We also design the optical marker add-ons for strut modules and node modules, containing five optical markers with a diameter of $1.9\,\mathrm{cm}$ at a distance of $4\,\mathrm{cm}$ or $5\,\mathrm{cm}$ from its origin, as shown in Fig. 12(b, c).

B. Localization Model Training

We collect a training dataset with 4800 data samples using a single strut module moved on the node base, as shown in Fig. 14(a). The strut module is manually controlled to move on the base. Each data sample includes the magnetic data and the estimated position of the magnet array center from the motion capture. The estimated positions are then transformed to the SAC by Eq. (1) as the ground truth.

The network is trained with the Adam optimizer, a batch size of 128, epochs of 300, and learning rate decay. The

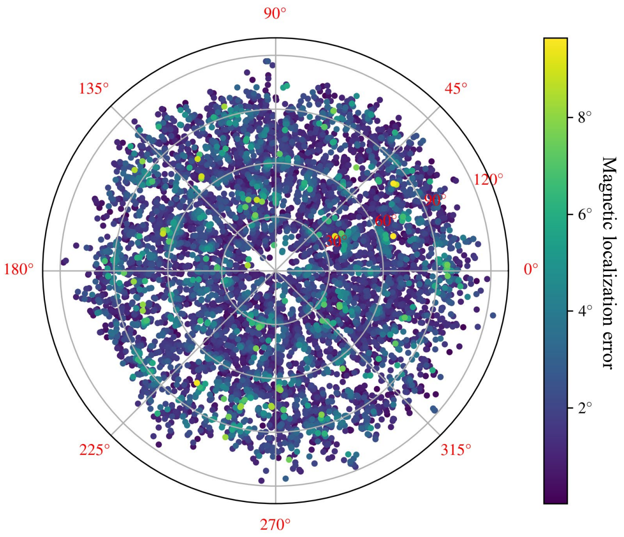

Fig. 13. Magnetic localization error distribution of the evaluation dataset in the polar coordinate system. The high error data points lie above the low error data points.

training only takes two minutes on an NVIDIA GeForce RTX 2080 Ti GPU. The central angle between the magnetic localization position vector and the ground truth is used as the metric. The root-mean-square (RMS) of the central angle error reaches 0.9° on the training dataset and 1.1° on a test dataset with 1000 data samples, and the classification accuracies both reach 100 % when the threshold value of existence probability is 0.9. The training and test datasets are collected together using the same modules.

A single target evaluation dataset is collected with three different strut modules automatically, including 7750 samples randomly selected mainly in the upper hemisphere. The RMS of central angle error is 2.2°, and the classification accuracy reaches 100 %. The localization error of this dataset is shown in Fig. 13.

C. Single Strut Module

In this experiment, we use the same setup as dataset collection and evaluate the accuracy of magnetic localization and strut orientation filter. The controller in Section V-A controls the strut module to follow a set of pre-defined waypoints.

Here, the strut module follows a spherical helix. As shown in Fig. 14(b), the magnetic localization result ${}^{H_i}\overline{\boldsymbol{z}}_{S_j}$ and the estimated z-axis ${}^{H_i}\hat{\boldsymbol{z}}_{S_j}$ are plotted. The RMS values of the central angle errors are 2.4° and 1.6°, respectively. The error of ${}^{H_i}\hat{\boldsymbol{z}}_{S_j}$ is smaller since it combines more sensor data.

The strut orientation from the motion capture and orientation filter can be compared based on the geodesic on the unit sphere [60], denoted as the geodesic error in the rest of this article. The RMS of the geodesic error is 2.0°. The orientations can also be compared as ZYX Euler angles, as plotted in Fig. 14(c). The localization errors are summarized in Table II.

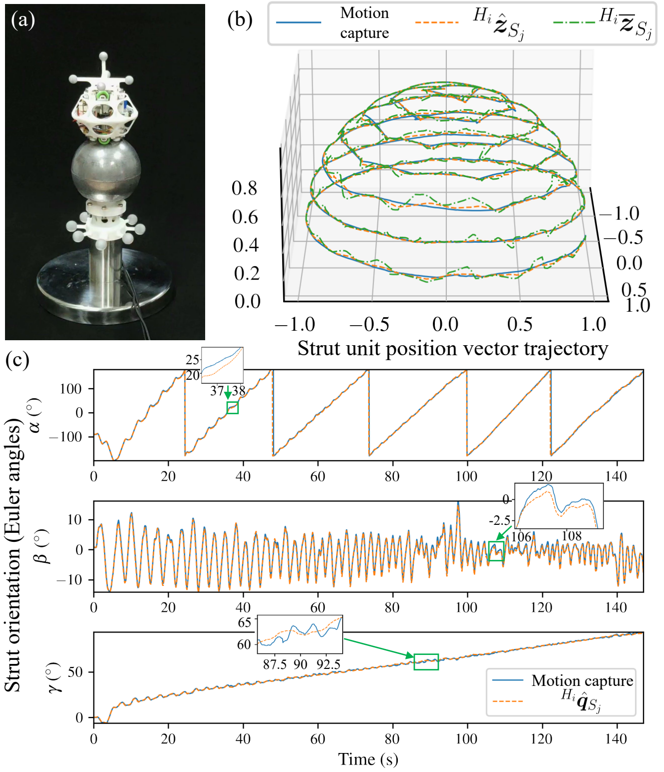

Fig. 14. Single strut accuracy evaluation. (a) Experimental setup. (b) Unit position vector trajectories of the strut. (c) Strut orientation as Euler angles.

TABLE II
SINGLE STRUT LOCALIZATION ERROR.

		Mean	RMS
Unit position vector central angle error (°)	${}^{H_i}\overline{\boldsymbol{z}}_{S_j}$	2.1	2.4
	${}^{H_i}\hat{\boldsymbol{z}}_{S_j}$	1.3	1.6
Strut orientation error (°)	Geodesic	1.8	2.0
	Euler angles α	1.4	1.7
	Euler angles β	0.7	1.0
	Euler angles γ	0.4	0.6

D. Multiple Strut Modules

The magnetic localization accuracy may vary with the number of connected strut modules since the magnetic fields generated by two magnet arrays can interfere with each other. In this experiment, we evaluate localization accuracy when five struts connect to the base, as shown in Fig. 15(a). The central strut module and the other four struts are connected to the base in sequence manually. Then, the central strut keeps still, while the other four struts are automatically identified and follow the pre-defined waypoints.

The variation of the central strut magnetic localization result is shown in Fig. 15(b), where the vertical coordinate is the central angle between the real-time magnetic localization result and the result after the first connection. The magnetic localization result varies less than 0.6° as the four struts move while the classification accuracy remains 100 %. And the RMS of the variation is 0.3°.

Fig. 15(c) shows part of the trajectories of the four struts,

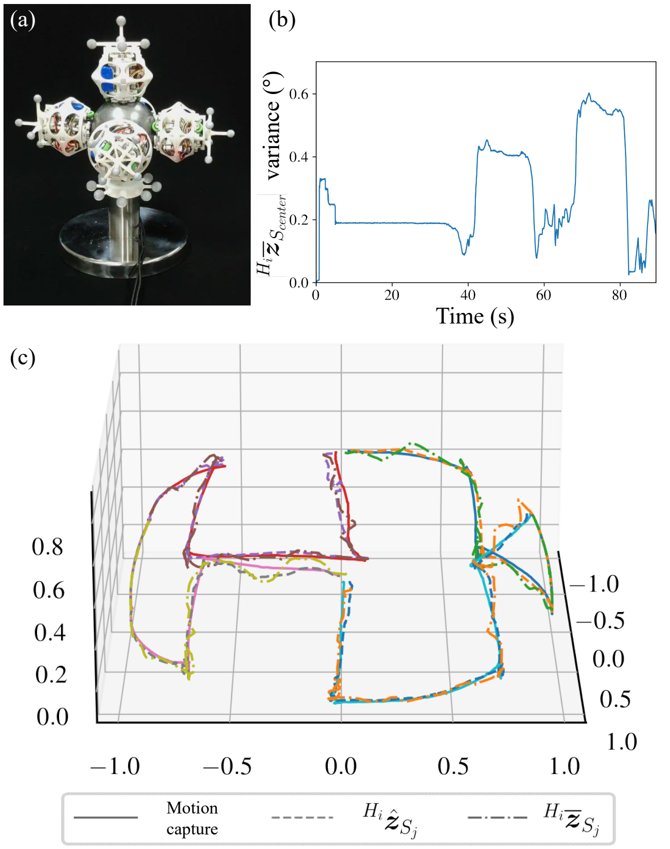

Fig. 15. Multiple struts accuracy evaluation. (a) Experimental setup. (b) Magnetic localization position vector variation of the central strut module as the other four struts connect and move. (c) Magnetic localization position vector trajectories of the other four struts.

including the magnetic localization result, estimated z-axis direction, and the motion capture ground truth with different line styles. The trajectories of the four struts are plotted with different colors, and the localization accuracy is summarized in Table III. When multiple struts connect to a node simultaneously, the overall localization error slightly increases since the magnet arrays can interfere with each other, but the effect is relatively small.

TABLE III
MULTIPLE STRUTS LOCALIZATION ERROR

	Strut1	Strut2	Strut3	Strut4	Overall
Magnetic localization central angle error (RMS, °)	2.6	2.9	2.7	2.3	2.6
Strut orientation geodesic error (RMS, °)	2.2	2.8	2.2	2.4	2.4

E. Relative Pose between Node Modules

In this experiment, the relative orientation between two adjacent node modules is evaluated, representing the estimation accuracy of the yaw angle difference filter. The effect of configuration optimization is also preliminarily evaluated.

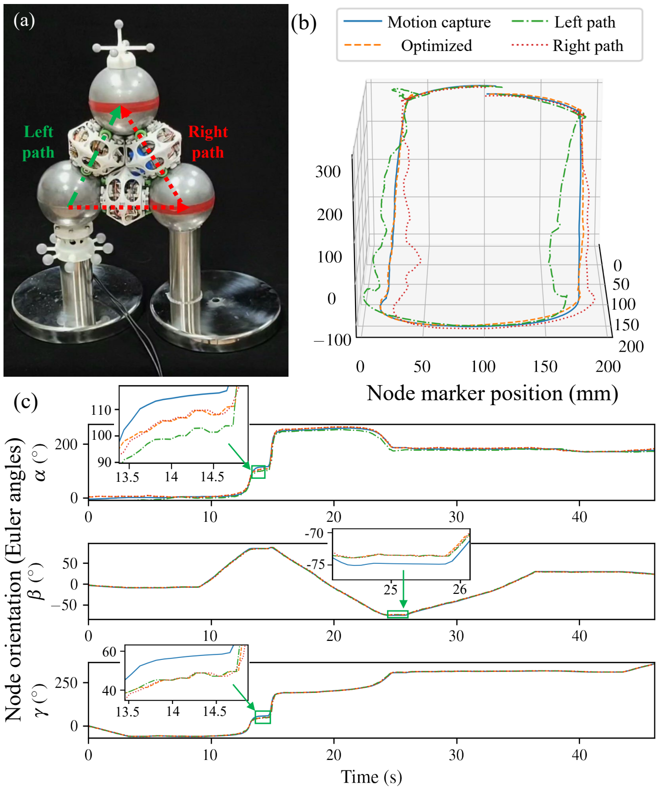

Fig. 16. Node relative pose accuracy evaluation. (a) Experimental setup. (b) Node marker position trajectories. (c) Node orientation as Euler angles.

As shown in Fig. 16(a), three nodes and three struts form a triangle structure. The two node modules below are fixed, and the pose of the node module above is changed by the strut modules. The base is the root module, and the node marker pose represented in the root node frame can be estimated based on Alg. 1 from the left path or the right path. Configuration optimization is also applied to have more accurate estimations using the Levenberg-Marquardt algorithm (LMA).

The estimated node marker position trajectories are compared with the ground truth in Fig. 16(b). The RMS values of Euclidean distance errors are 13.3 mm and 11.0 mm from the left path and the right path, respectively. After optimization, the RMS error reduces to 6.0 mm.

The node marker orientations in Euler angles representation are plotted in Fig. 16(c), and the localization errors are summarized in Table IV.

TABLE IV
NODE RELATIVE POSE ERROR

	Left path	Right path	Optimized
Node marker position Euclidean error (RMS, mm)	13.3	11.0	6.0
Node orientation geodesic error (RMS, °)	6.0	4.2	4.0

VII. SYSTEM BEHAVIOR DEMONSTRATION

In this section, we demonstrate the automation of two behaviors in the library, where seven strut modules and four node

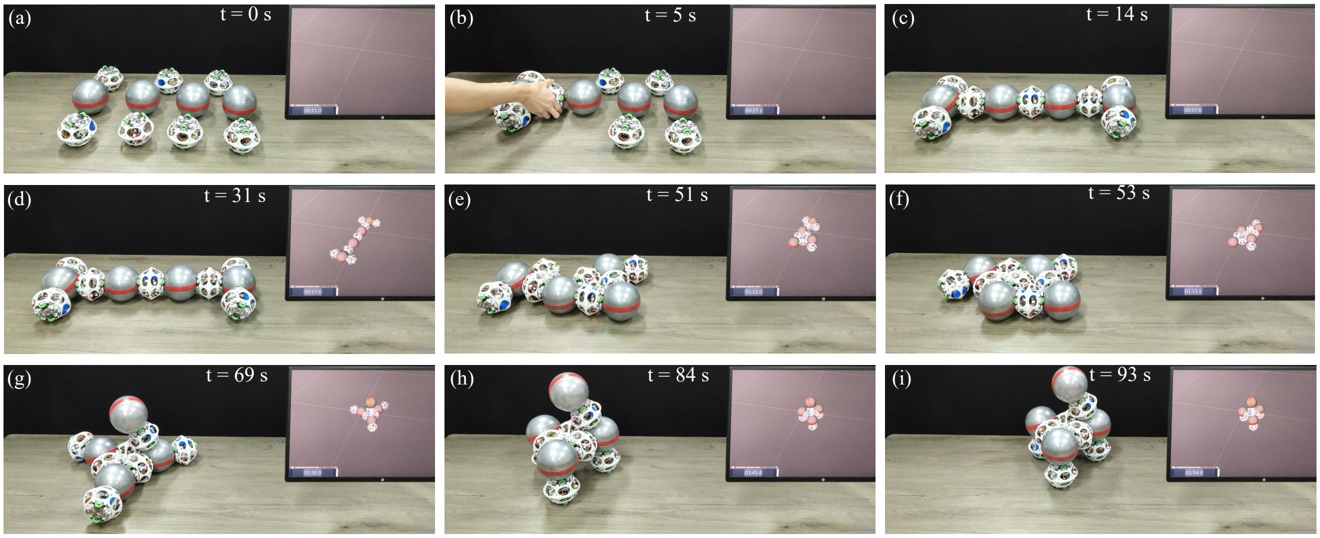

Fig. 17. Self-reconfiguration. (a) - (c): Seven strut modules and four node modules are assembled manually, and the magnetic localization and identification start automatically. (d) - (g): The modules self-reconfigure to a three-legged vehicle configuration. (h) - (i): The legged vehicle stands up and moves forward using the wheels.

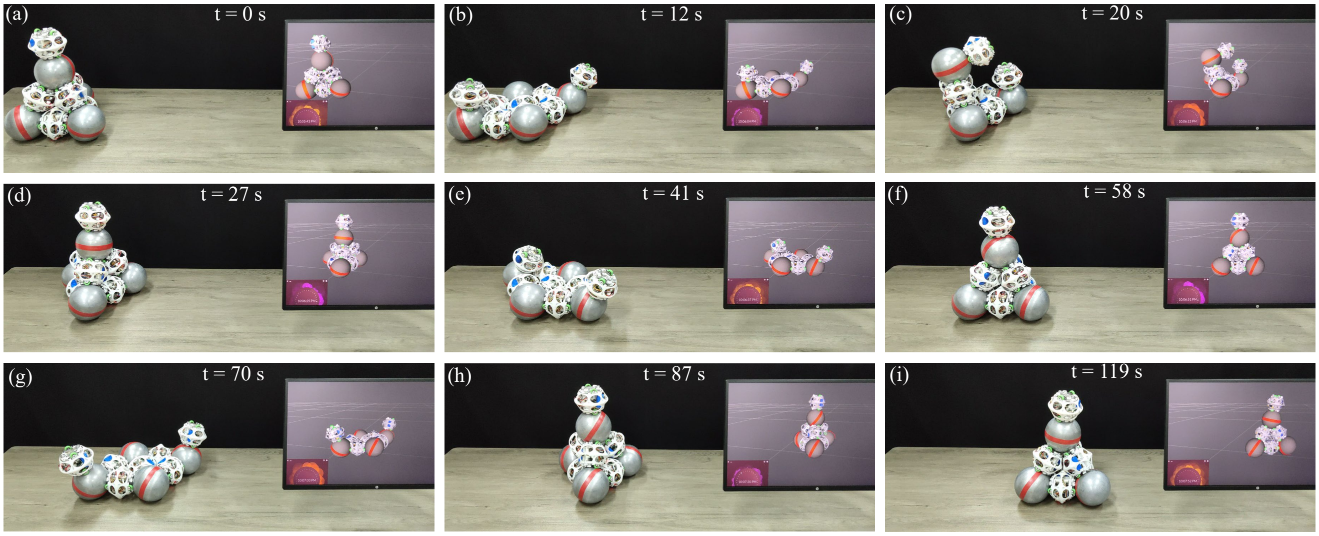

Fig. 18. Rolling. (a): Seven strut modules and four node modules form the tetrahedron configuration. (b) - (c): The modules reconfigure to the double-arm configuration and lower the front arm. (d): The modules raise the rear arm and reconfigure back to the tetrahedron configuration. (e) - (i): The tetrahedron rolls forward by repeating the sequence of motions.

modules are used. The magnetic localization, module filters, and controllers run distributed on the microcontrollers inside the FreeSN modules. The system configuration is estimated in a centralized computer and visualized on the screen with RViz in real time.

A. Self-reconfiguration

As shown in Fig. 17, the FreeSN modules are assembled manually initially. The system configuration is automatically identified after the magnetic module identification process, and the modules execute the self-reconfiguration behavior. The modules converge towards the center, completing two connections and one disconnection to form the three-legged vehicle configuration. The connection motion is fast since the magnetic attraction helps to align the connector, and magnetic module identification is no longer needed. At $t = 69\,\mathrm{s}$, the three-legged vehicle stands up and then moves forward using the wheels.

B. Rolling

As shown in Fig. 18(a), the modules form the tetrahedron configuration initially. This configuration can roll forward by executing a series of motions. First, the tetrahedron reconfigures to the double-arm configuration by a disconnection, as shown in Fig. 18(b). Then, modules lower the front arm and raise the rear arm while keeping the center of mass stable by adjusting the position of the top two struts. Finally, the modules reconfigure back to the tetrahedron configuration after the connection, as shown in Fig. 18(d). The modules roll forward four times by repeating the motion sequence and reach the center of the table, as shown in Fig. 18(i).

VIII. Discussions and Future Work

Accurate position sensing is important in the field of robotics. This article presents a complete configuration identification system for FreeSN, including connection point magnetic localization, module identification, module orientation fusion, and system configuration fusion. With the sensing feedback, a shortest-arc controller of the freeform differential driving vehicle is also proposed. The FreeSN modules can execute the control task sequence automatically based on the library, which is validated by self-reconfiguration and rolling behaviors. Compared to the previous manual control demonstrations [24], the modules move more smoothly and with greater parallelism, representing the real speed of the current FreeSN system. The sensing accuracy is also evaluated with the motion capture system. The systems can be easily migrated to other freeform MSRRs [22], [23] based on magnetic connections.

The magnet array of the presented module is highly symmetric since the modules are designed to be compact for better mechanical performance. However, this makes it difficult to accurately estimate the orientation of the magnet array only by magnetic localization. The proposed strut orientation filter estimates the strut orientation by fusing the magnetic localization position vector, accelerometer, gyroscope, and wheel odometry. When the angle between the magnetic localization position vector and gravity is small, this 9-axis orientation filter will degrade to a 6-axis orientation filter. The filter can drift if the strut spins in this condition for a long time, although such motion demands rarely occur for most applications.

The presented configuration identification system provides the foundation to explore this freeform and strut-node structured MSRR system. The current system relies on a centralized controller to estimate the global configuration and assign and synchronize the distributed control tasks. The scalability of the system is limited by the software architecture in large-scale applications, where distributed control is essential for robustness and scalability [61]. We are interested in investigating the advantages of the distributed FreeSN system for unstructured environment crossing tasks without external sensors. In future work, visual sensors will be integrated into the strut modules for localization between robot groups and environmental perception. The distributed configuration planning and control algorithm under gravity stability [27] also needs to be further investigated. Another promising research direction is a higher dimension and more accurate magnetic localization system. Both magnetic sensor array and magnet array arrangements can be improved while reserving mechanical performance.

Appendix A
Derivations of Strut Orientation Filter

A. Simplified Gradient of Strut Orientation Filter

Obviously, ${}^{H_i}\hat{\boldsymbol{g}} = {}^{S_j}\hat{\boldsymbol{z}}_{S_j} = \begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T$, and we can simplify the objective functions as:

$$f_g({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, \hat{\boldsymbol{a}}_{S_j}) = \begin{bmatrix}2(\epsilon_1\epsilon_3 - \eta\epsilon_2) \\ 2(\epsilon_2\epsilon_3 + \eta\epsilon_1) \\ 1 - 2\epsilon_1^2 - 2\epsilon_2^2\end{bmatrix} - \hat{\boldsymbol{a}}_{S_j} \tag{27}$$

$$f_z({}^{H_i}_{S_j}\hat{\boldsymbol{q}}, {}^{H_i}\overline{\boldsymbol{z}}_{S_j}) = \begin{bmatrix}2(\epsilon_1\epsilon_3 + \eta\epsilon_2) \\ 2(\epsilon_2\epsilon_3 - \eta\epsilon_1) \\ 1 - 2\epsilon_1^2 - 2\epsilon_2^2\end{bmatrix} - {}^{H_i}\overline{\boldsymbol{z}}_{S_j} \tag{28}$$

where η and $\boldsymbol{\epsilon} = \begin{bmatrix}\epsilon_1 & \epsilon_2 & \epsilon_3\end{bmatrix}^T$ are the scalar and vector parts of the quaternion ${}^{H_i}_{S_j}\hat{\boldsymbol{q}} = \begin{bmatrix}\eta & \boldsymbol{\epsilon}\end{bmatrix}^T \in \mathbb{R}^4$, respectively. The Jacobian matrices of the objective functions can also be simplified as:

$$J_g({}^{H_i}_{S_j}\hat{\boldsymbol{q}}) = \begin{bmatrix}-2\epsilon_2 & 2\epsilon_3 & -2\eta & 2\epsilon_1 \\ 2\epsilon_1 & 2\eta & 2\epsilon_3 & 2\epsilon_2 \\ 0 & -4\epsilon_1 & -4\epsilon_2 & 0\end{bmatrix} \tag{29}$$

$$J_z({}^{H_i}_{S_j}\hat{\boldsymbol{q}}) = \begin{bmatrix}2\epsilon_2 & 2\epsilon_3 & 2\eta & 2\epsilon_1 \\ -2\epsilon_1 & -2\eta & 2\epsilon_3 & 2\epsilon_2 \\ 0 & -4\epsilon_1 & -4\epsilon_2 & 0\end{bmatrix} \tag{30}$$

Finally, the gradient ∇f can be calculated by Eq. (12).

B. Derivation of Strut Relative Angular Velocity Estimation

Based on the definition of frame ΣH_i, we have

$$\begin{cases}{}^{H_i}_{E}\boldsymbol{\omega}_{H_i} \circ \begin{bmatrix}1 & 1 & 0\end{bmatrix}^T = \mathbf{0} \\ {}^{H_i}_{H_i}\boldsymbol{\omega}_{N_i} \circ \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T = \mathbf{0}\end{cases} \tag{31}$$

Then we have

$$\begin{aligned}{}^{H_i}_{H_i}\overline{\boldsymbol{\omega}}_{S_j} = {}^{H_i}_{H_i}\boldsymbol{\omega}^{gyro}_{S_j} \circ \begin{bmatrix}1 \\ 1 \\ 0\end{bmatrix} + {}^{H_i}_{H_i}\boldsymbol{\omega}^{go}_{S_j} \circ \begin{bmatrix}0 \\ 0 \\ 1\end{bmatrix} = \\ \left({}^{H_i}_{E}\boldsymbol{\omega}^{gyro}_{S_j} - {}^{H_i}_{E}\boldsymbol{\omega}^{gyro}_{H_i}\right) \circ \begin{bmatrix}1 & 1 & 0\end{bmatrix}^T + \\ \left({}^{H_i}_{H_i}\boldsymbol{\omega}^{go}_{N_i} + {}^{H_i}_{N_i}\boldsymbol{\omega}^{go}_{S_j}\right) \circ \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T = \\ {}^{H_i}_{E}\boldsymbol{\omega}^{gyro}_{S_j} \circ \begin{bmatrix}1 \\ 1 \\ 0\end{bmatrix} + {}^{H_i}_{N_i}\boldsymbol{\omega}^{go}_{S_j} \circ \begin{bmatrix}0 \\ 0 \\ 1\end{bmatrix}\end{aligned} \tag{32}$$

Since ${}^{S_j}_{H_i}\overline{\boldsymbol{\omega}}_{S_j} = {}^{H_i}_{S_j}\boldsymbol{R}^T\, {}^{H_i}_{H_i}\overline{\boldsymbol{\omega}}_{S_j}$, ${}^{H_i}_{E}\boldsymbol{\omega}^{gyro}_{S_j} = {}^{H_i}_{S_j}\boldsymbol{R}\ \boldsymbol{\omega}^{gyro}_{S_j}$, and ${}^{H_i}_{N_i}\boldsymbol{\omega}^{go}_{S_j} = {}^{H_i}_{S_j}\boldsymbol{R}\, {}^{S_j}_{N_i}\boldsymbol{\omega}^{go}_{S_j}$, we can get ${}^{S_j}_{H_i}\overline{\boldsymbol{\omega}}_{S_j}$ as described in Eq. (15).

References

[1] A. K. S. Sankhar Reddy Chennareddy, Anita Agrawal, "Modular self-reconfigurable robotic systems: A survey on hardware architectures," *Journal of Robotics*, vol. 2017, p. 1–19, 2017.

[2] M. Yim, W. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. S. Chirikjian, "Modular self-reconfigurable robot systems [grand challenges of robotics]," *IEEE Robotics Automation Magazine*, vol. 14, no. 1, pp. 43–52, 2007.

[3] J. Seo, J. Paik, and M. Yim, "Modular reconfigurable robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, no. 1, pp. 63–88, 2019. [Online]. Available: https://doi.org/10.1146/annurev-control-053018-023834

[4] A. Brunete, A. Ranganath, S. Segovia, J. P. de Frutos, M. Hernando, and E. Gambao, "Current trends in reconfigurable modular robots design," *International Journal of Advanced Robotic Systems*, vol. 14, no. 3, p. 1729881417710457, 2017. [Online]. Available: https://doi.org/10.1177/1729881417710457

[5] A. Hayat, “A framework for taxonomy and evaluation of self-reconfigurable robotic systems,” *IEEE Access*, 01 2020.

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

[7] A. Spröwitz, S. Pouya, S. Bonardi, J. V. Den Kieboom, R. Möckel, A. Billard, P. Dillenbourg, and A. J. Ijspeert, “Roombots: Reconfigurable robots for adaptive furniture,” *IEEE Computational Intelligence Magazine*, vol. 5, no. 3, pp. 20–32, 2010.

[8] C. H. Belke and J. Paik, “Mori: A modular origami robot,” *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 5, pp. 2153–2164, 2017.

[9] M. W. Jorgensen, E. H. Ostergaard, and H. H. Lund, “Modular ATRON: modules for a self-reconfigurable robot,” in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 2, 2004, pp. 2068–2073 vol.2.

[10] H. Wei, Y. Chen, J. Tan, and T. Wang, “Sambot: A self-assembly modular robot system,” *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 4, pp. 745–757, 2011.

[11] V. Zykov, E. Mytilinaios, M. Desnoyer, and H. Lipson, “Evolved and designed self-reproducing modular robotics,” *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 308–319, 2007.

[12] J. W. Romanishin, K. Gilpin, S. Claici, and D. Rus, “3D M-Blocks: Self-reconfiguring robots capable of locomotion via pivoting in three dimensions,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1925–1932.

[13] J. Davey, N. Kwok, and M. Yim, “Emulating self-reconfigurable robots - design of the SMORES system,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 4464–4469.

[14] T. Tosun, J. Davey, C. Liu, and M. Yim, “Design and characterization of the EP-Face connector,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 45–51.

[15] J. Neubert, A. Rost, and H. Lipson, “Self-soldering connectors for modular robots,” *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1344–1357, 2014.

[16] P. Swissler and M. Rubenstein, “FireAnt3D: a 3d self-climbing robot towards non-latticed robotic self-assembly,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 3340–3347.

[17] R. F. M. Garcia, J. D. Hiller, K. Stoy, and H. Lipson, “A vacuum-based bonding mechanism for modular robotics,” *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 876–890, 2011.

[18] N. Eckenstein and M. Yim, “Area of acceptance for 3d self-aligning robotic connectors: Concepts, metrics, and designs,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1227–1233.

[19] M. Shimizu, T. Mori, and A. Ishiguro, “A development of a modular robot that enables adaptive reconfiguration,” in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 174–179.

[20] H. Tokashiki, H. Amagai, S. Endo, K. Yamada, and J. Kelly, “Development of a transformable mobile robot composed of homogeneous gear-type units,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 2, 2003, pp. 1602–1607 vol.2.

[21] J. Campbell, P. Pillai, and S. Goldstein, “The robot is the tether: active, adaptive power routing modular robots with unary inter-robot connectors,” in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 4108–4115.

[22] G. Liang, H. Luo, M. Li, H. Qian, and T. L. Lam, “FreeBOT: A freeform modular self-reconfigurable robot with arbitrary connection point - design and implementation,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 6506–6513.

[23] D. Zhao and T. L. Lam, “SnailBot: A continuously dockable modular self-reconfigurable robot using rocker-bogie suspension,” in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 4261–4267.

[24] Y. Tu, G. Liang, and T. L. Lam, “FreeSN: A freeform strut-node structured modular self-reconfigurable robot - design and implementation,” in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 4239–4245.

[25] M. Malley, B. Haghighat, L. Houel, and R. Nagpal, “Eciton robotica: Design and algorithms for an adaptive self-assembling soft robot collective,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4565–4571.

[26] P. Swissler and M. Rubenstein, “Reactivebuild: Environment-adaptive self-assembly of amorphous structures,” in *Distributed Autonomous Robotic Systems*, F. Matsuno, S.-i. Azuma, and M. Yamamoto, Eds. Cham: Springer International Publishing, 2022, pp. 363–375.

[27] H. Luo and T. L. Lam, “Adaptive flow planning of modular spherical robot considering static gravity stability,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4228–4235, 2022.

[28] C. Liu, T. Tosun, and M. Yim, “A low-cost, highly customizable solution for position estimation in modular robots,” *Journal of Mechanisms and Robotics*, 2021.

[29] S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji, “M-TRAN: self-reconfigurable modular robotic system,” *IEEE/ASME Transactions on Mechatronics*, vol. 7, no. 4, pp. 431–441, 2002.

[30] H. Kurokawa, K. Tomita, A. Kamimura, S. Kokaji, T. Hasuo, and S. Murata, “Distributed self-reconfiguration of M-TRAN III modular robotic system,” *The International Journal of Robotics Research*, vol. 27, no. 3-4, pp. 373–386, 2008.

[31] M. Yim, D. Duff, and K. Roufas, “PolyBot: a modular reconfigurable robot,” in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, 2000, pp. 514–520 vol.1.

[32] D. Rus and M. Vona, “Crystalline robots: Self-reconfiguration with compressible unit modules,” *Autonomous Robots*, vol. 10, no. 1, pp. 107–124, Jan 2001. [Online]. Available: https://doi.org/10.1023/A:1026504804984

[33] K. Abe, K. Tadakuma, and R. Tadakuma, “ABENICS: Active ball joint mechanism with three-dof based on spherical gear meshings,” *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1806–1825, 2021.

[34] K.-M. Lee and D. Zhou, “A spherical encoder for real-time measurements of three-DOF wrist orientations,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 2, 2003, pp. 1596–1601 vol.2.

[35] L. Yan, I.-M. Chen, Z. Guo, Y. Lang, and Y. Li, “A three degree-of-freedom optical orientation measurement method for spherical actuator applications,” *IEEE Transactions on Automation Science and Engineering*, vol. 8, no. 2, pp. 319–326, 2011.

[36] L. Yan, B. Zhu, Z. Jiao, C.-Y. Chen, and I.-M. Chen, “An orientation measurement method based on hall-effect sensors for permanent magnet spherical actuators with 3d magnet array,” *Scientific Reports*, vol. 4, no. 1, p. 6756, Oct 2014. [Online]. Available: https://doi.org/10.1038/srep06756

[37] 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*, 2008, pp. 883–888.

[38] A. Spinos, D. Carroll, T. Kientz, and M. Yim, “Variable topology truss: Design and analysis,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2717–2722.

[39] N. Melenbrink, P. Michalatos, P. Kassabian, and J. Werfel, “Using local force measurements to guide construction by distributed climbing robots,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 4333–4340.

[40] J. Liu, H. Sugiyama, T. Nakayama, and S. Miyashita, “Magnetic Sensor Based Topographic Localization for Automatic Dislocation of Ingested Button Battery,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 5488–5494, iSSN: 2577-087X.

[41] S. Song, C. Hu, and M. Q.-H. Meng, “Multiple Objects Positioning and Identification Method Based on Magnetic Localization System,” *IEEE Transactions on Magnetics*, vol. 52, no. 10, pp. 1–4, Oct. 2016, conference Name: IEEE Transactions on Magnetics.

[42] M. Gherardini, F. Clemente, S. Milici, and C. Cipriani, “Localization accuracy of multiple magnets in a myokinetic control interface,” *Scientific Reports*, vol. 11, no. 1, p. 4850, Mar 2021. [Online]. Available: https://doi.org/10.1038/s41598-021-84390-8

[43] D. Son, S. Yim, and M. Sitti, “A 5-d localization method for a magnetically manipulated untethered robot using a 2-d array of hall-effect sensors,” *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 2, pp. 708–716, 2016.

[44] D. Son, X. Dong, and M. Sitti, “A simultaneous calibration method for magnetic robot localization and actuation systems,” *IEEE Transactions on Robotics*, vol. 35, no. 2, pp. 343–352, 2019.

[45] D. Cichon, R. Psiuk, H. Brauer, and H. Töpfer, “A Hall-Sensor-Based Localization Method With Six Degrees of Freedom Using Unscented Kalman Filter,” *IEEE Sensors Journal*, vol. 19, no. 7, pp. 2509–2516, Apr. 2019, conference Name: IEEE Sensors Journal.

[46] C. Di Natali, M. Beccani, N. Simaan, and P. Valdastri, “Jacobian-Based Iterative Method for Magnetic Localization in Robotic Capsule

Endoscopy," *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 327–338, Apr. 2016, conference Name: IEEE Transactions on Robotics.

[47] S. Chang, Y. Lin, Y. R. Zheng, and X. Fu, "Simultaneous detection of multiple magnetic dipole sources," *IEEE Transactions on Magnetics*, vol. 56, no. 9, pp. 1–11, 2020.

[48] N. Sebkhi, N. Sahadat, S. Hersek, A. Bhavsar, S. Siahpoushan, M. Ghoovanloo, and O. T. Inan, "A Deep Neural Network-Based Permanent Magnet Localization for Tongue Tracking," *IEEE Sensors Journal*, vol. 19, no. 20, pp. 9324–9331, Oct. 2019, conference Name: IEEE Sensors Journal.

[49] F. Wu, N. M. Robert, D. D. Frey, and S. Foong, "Enhanced magnetic localization with artificial neural network field models," in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 1560–1565, iSSN: 1050-4729.

[50] F. Y. Wu, S. Foong, and Z. Sun, "A hybrid field model for enhanced magnetic localization and position control," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 3, pp. 1278–1287, 2015.

[51] R. Yu, S. L. Charreyron, Q. Boehler, C. Weibel, C. Chautems, C. C. Y. Poon, and B. J. Nelson, "Modeling electromagnetic navigation systems for medical applications using random forests and artificial neural networks," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9251–9256.

[52] C. Watson and T. K. Morimoto, "Permanent Magnet-Based Localization for Growing Robots in Medical Applications," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2666–2673, Apr. 2020, conference Name: IEEE Robotics and Automation Letters.

[53] S. L. Charreyron, Q. Boehler, B. Kim, C. Weibel, C. Chautems, and B. J. Nelson, "Modeling electromagnetic navigation systems," *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1009–1021, 2021.

[54] Y. Tu, G. Liang, and T. L. Lam, "Graph convolutional network based configuration detection for freeform modular robot using magnetic sensor array," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 4252–4258.

[55] K. Lei, D. Qi, and X. Tian, "A new coordinate system for constructing spherical grid systems," *Applied Sciences*, vol. 10, no. 2, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/2/655

[56] S. O. H. Madgwick, A. J. L. Harrison, and R. Vaidyanathan, "Estimation of imu and marg orientation using a gradient descent algorithm," in *2011 IEEE International Conference on Rehabilitation Robotics*, 2011, pp. 1–7.

[57] J. Borenstein and L. Feng, "Gyrodometry: a new method for combining data from gyros and odometry in mobile robots," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, 1996, pp. 423–428 vol.1.

[58] G. Welch, G. Bishop *et al.*, "An introduction to the kalman filter," 1995.

[59] K. Paton, "An algorithm for finding a fundamental set of cycles of a graph," *Commun. ACM*, vol. 12, pp. 514–518, 1969.

[60] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, pp. 155–164, 2009.

[61] Z. Butler, K. Kotay, D. Rus, and K. Tomita, "Generic decentralized control for lattice-based self-reconfigurable robots," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 919–937, 2004.

Yuxiao Tu (Student Member, IEEE) received the B.Eng. (First Class Hons.) degree in electronic and information engineering in 2020 from The Chinese University of Hong Kong, Shenzhen, where he is currently working toward the Ph.D. degree in computer and information engineering.

His current research interests include modular robotics and reconfigurable robotics.

Tin Lun Lam (Senior Member, IEEE) received the B.Eng. (First Class Hons.) and Ph.D. degrees in robotics and automation from the Chinese University of Hong Kong, Hong Kong, in 2006 and 2010, respectively.

He is currently an Assistant Professor with the Chinese University of Hong Kong, Shenzhen, China, the Executive Deputy Director of the National-Local Joint Engineering Laboratory of Robotics and Intelligent Manufacturing, and the Director of Center for the Intelligent Robots, Shenzhen Institute of Artificial Intelligence and Robotics for Society. His research interests include multirobot systems, field robotics, and collaborative robotics.