Graph Convolutional Network based Configuration Detection for Freeform Modular Robot Using Magnetic Sensor Array

Yuxiao Tu[1,2], Guanqi Liang[1,2], and Tin Lun Lam[1,2,†]

***Abstract*— Modular self-reconfigurable robotic (MSRR) systems are potentially more robust and more adaptive than conventional systems. Following our previous work where we proposed a freeform MSRR module called FreeBOT, this paper presents a novel configuration detection system for FreeBOT using a magnetic sensor array. A FreeBOT module can be connected by up to 11 modules, and the proposed configuration detection system can locate a variable number of connection points accurately in real-time. By equipping FreeBOT with 24 magnetic sensors, the magnetic field density produced by magnets and steel spherical shells can be monitored. The connectable area is split into 199 non-uniform regions, including 84 uniform regions. Using a Graph Convolutional Network (GCN) based algorithm, the connection points can be located accurately under ferromagnetic environments. The system can locate a variable number of connection points for such a region division with only single connection point training data. Finally, the localization algorithm can run faster than 40 Hz on FreeBOT. With the real-time configuration detection system, the FreeBOT system has the potential to reconfigure automatically and accurately.**

I. INTRODUCTION

Modular self-reconfigurable robotic (MSRR) system [1]–[3] can rearrange its repeated modules into different configurations, in order to adapt to new circumstances, perform new tasks, or recover from damage. Many previous MSRR modules [4]–[8] have been designed, but they are hard to construct a freeform robotic system due to their physical constraints.

In our previous work, we present a freeform MSRR called FreeBOT (Freeform Robot) [9], [10]. The FreeBOT can move independently, connect/disconnect from other FreeBOT, and move on other FreeBOT freely. A FreeBOT system can reconfigure to different configurations freely and has the potential to realize a freeform robotic system. To realize automatic reconfiguration, a configuration detection system without external sensors is needed as the feedback for reconfiguration control. The system needs to locate all connection points accurately through a 0.5mm thick low-carbon steel spherical shell. However, sensors can hardly be placed at the steel shell surface limited by the free movement of the internal vehicle. An electronic compass does not work due to the unknown and variational magnet magnetic field and steel remanence. Most modern low-power sensors such as optical, capacitive, and inductive sensors are blocked by the steel shell. The magnet inside the FreeBOT provides a strong magnetic force, and the steel shell cannot block such a strong signal when the FreeBOTs are connected. A magnetic localization system has the potential to detect the FreeBOT configuration accurately.

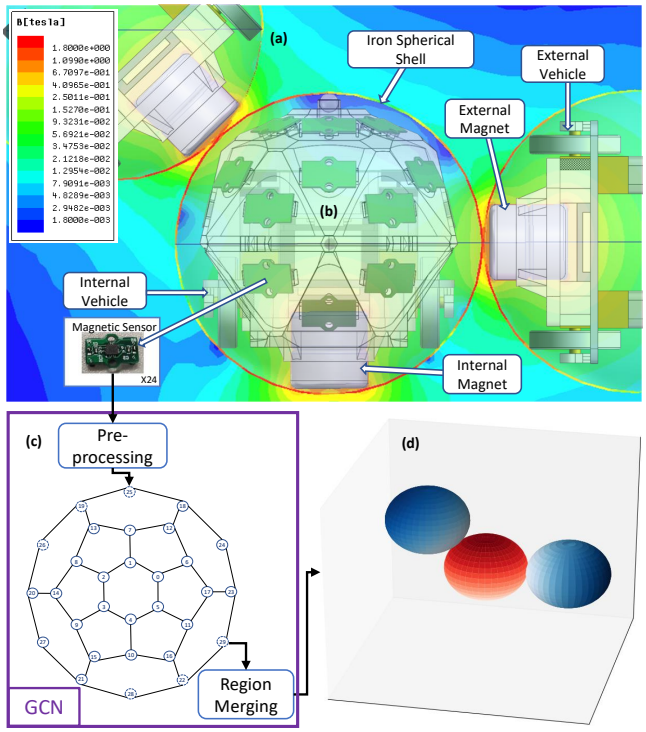

Fig. 1. Overview of the proposed configuration detection system. (a) Magnetic field distribution by Ansys Maxwell simulator when two FreeBOTs connect to a FreeBOT. (b) FreeBOT equipped with a magnetic sensor array. (c) Graph convolutional network-based localization algorithm, which estimates the locations of multiple connection points with magnetic sensor data. (d) Real-time configuration visualization.

The permanent magnet localization method has been widely studied in the past decades, which is very popular for biomedical applications. Many model-based localization algorithms model magnet behaves as a magnetic dipole, and various algorithms such as trilateration [11], Levenberg-Marquardt algorithm (LMA) [12], Trust Region Reflective algorithm (TRRA) [13], Unscented Kalman Filter (UKF) [14], and jacobian-based iterative method [15] are proposed to solve the multiple nonlinear targets tracking problem. Due to the inaccuracy of the dipole at the surface of the magnet, some Artificial Neural Network (ANN) based [16]–[19] and hybrid [20] algorithms are proposed to approximate the magnetic field and achieve good localization accuracy when

*This work was supported by the National Natural Science Foundation of China (62073274), and the funding 2019-INT008 from the Shenzhen Institute of Artificial Intelligence and Robotics for Society.

[1]The Chinese University of Hong Kong, Shenzhen.
[2]Shenzhen Institute of Artificial Intelligence and Robotics for Society.
[†]Corresponding author is Tin Lun Lam tllam@cuhk.edu.cn

tracking a single magnet. However, ferromagnetic materials such as low-carbon steel can heavily change the magnetic field distribution, and the traditional magnetic models fail in such scenes. It is possible to approximate the magnetic field in ferromagnetic environments with deep learning models.

This paper presents a novel configuration detection system using a magnetic sensor array, as shown in Fig. 1. A Graph Convolutional Network (GCN) [21], [22] based algorithm is proposed to locate a variable number of connection points under ferromagnetic environments. The connection point locations are discretized and divided into 199 non-uniform sub-regions. The system shows good accuracy and robustness for multiple connection point localization trained on single connection point data. The proposed algorithm can run on a Quad-core Cortex™-A7 Linux-based controller at 40 Hz. Equipped with the configuration detection system, the FreeBOT has the potential to reconfigure automatically and accurately. The z-axes of the 6-axis Inertial Measurement Unit (IMU) in multiple FreeBOT modules can be synchronized by this system. The main contributions of the paper are:

1) to present the concept of configuration detection for a freeform modular robot (FreeBOT) with a magnetic sensor array and propose a magnetic sensor array arrangement for configuration detection under magnetic hysteresis interference.
2) to propose a Graph Convolutional Network-based multiple connection point localization algorithm under ferromagnetic environments. The network is trained with only single connection point data. This preliminarily tries to do magnetic localization with the neural network under a ferromagnetic environment.
3) to demonstrate the real-time configuration detection system, which provides necessary feedback for reconfiguration closed-loop control.

II. METHODS

This section will introduce the FreeBOT equipped with magnetic sensors, followed by the theoretic modeling of the magnetic localization problem and the GCN-based localization algorithm.

A. Magnetic Localization Module

The prototype of FreeBOT consists of a vehicle, an internal magnet, and a low-carbon steel spherical shell. The magnet produces a strong magnetic field, and the surrounding steel is magnetized. When a FreeBOT approaches the internal magnet of another FreeBOT, the magnetic attraction is generated, and the two FreeBOT are connected. In order to automate the FreeBOT self-reconfiguration, we build a magnetic sensor array to locate the connection points. As shown in Fig. 2, the sensor array contains 24 Melexis MLX90393 tri-axis magnetic sensors on the 3D printed framework of the internal vehicle. The sensors are connected by the flexible flat cable (FFC) and communicate with an Allwinner H3 (Quad-core Cortex™-A7) Linux-based controller through I2C-bus. The specifications of hardware can be found in Table I. The data

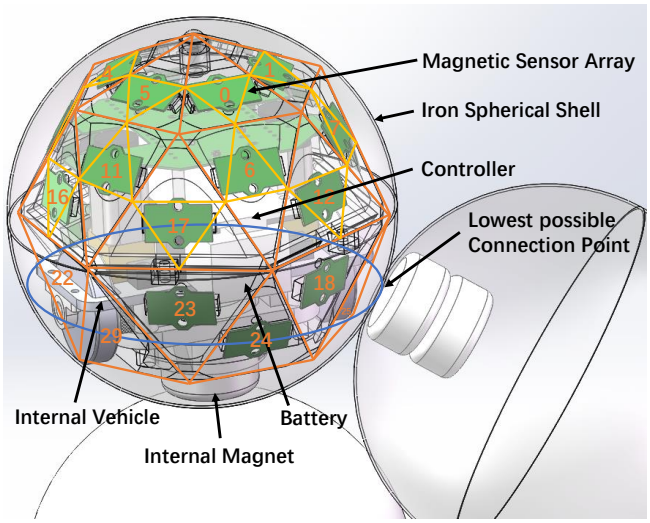

Fig. 2. FreeBOT that equipped with magnetic sensor array and region division. 24 magnetic sensors are arranged on the internal frame of FreeBOT, and the region division for our localization algorithm is plotted. Orange line: region divide line; Yellow line: sub-region divide line; Blue circle: connectable area divide line; Orange number: region id.

sampling rate is adjustable according to on-chip sampling and filtering configurations.

To eliminate magnetic field uncertainty caused by the magnetization history, most sensors are placed close to the steel shell. The magnetic field uncertainty is caused by the steel that is not magnetized to saturation. The magnetic field decreases super-quadratic as the distance from the magnetic source increases, and the superficial magnetic field of the external magnet is much larger than the superficial magnetic field of steel. To locate the connected external magnets free from steel remanence, the closest several magnetic sensors should be placed close to the external magnet, where is also far away from the unsaturated magnetized steel. Following this principle, the sensors are placed around 5 mm - 8 mm away from the steel shell. As the number of sensors increases, more sensors are free from steel remanence, and the sensor array can locate the connection points more accurately.

B. Configuration Detection Modeling

For a single FreeBOT, the purpose of reconfiguration localization is to detect the connection points of other FreeBOTs that connect to this robot. The measured magnetic data can be described as,

$$B_{sens} = f(\phi_1, \theta_1, ..., \phi_N, \theta_N),$$

where B_{sens} is a set of magnetic field density sampled inside the surrounding steel spherical shell, the polar angle ϕ_j and azimuth angle θ_j is the polar coordinates of the j_{th} connection point related to the internal vehicle, and N is the number of connection points. Once we know the magnetic field density measured by the sensors, it is possible to find

TABLE I

SPECIFICATIONS OF HARDWARE SETUP

Melexis MLX90393 Triaxis Manetic Sensor	24 counts
- X,Y Axis Sensitivity	3.004 uT/LSB
- Z Axis Sensitivity	4.840 uT/LSB
- X,Y Axis Measurement Range	±66.088 mT
- Z Axis Measurement Range	±106.480 mT
Low-carbon Steel Spherical Shell	
- External Diameter	80.0 mm
- Thickness	0.5 mm
- Saturation Magnetization	1.87 T
N52 NdFeB Cylinder Magnet	
- Diameter	20 mm
- Height	10 mm
- Superficial Magnetic Field	0.4696 T
- Remanence	1.37 T
- Maximum Energy Product	52 MGOe

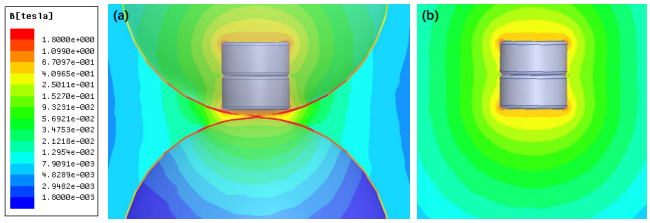

Fig. 3. Magneic field distribution with (a) and without (b) surrounding steel spherical shells.

f^{-1}, and we can localize the connection points by

$$(\phi_1, \theta_1, ..., \phi_N, \theta_N) = f^{-1}(B_{sens}),$$

A magnetic sensor perceives a magnetic field both from the internal magnet, external magnets, and magnetized steel as

$$B^i_{sens} = B_{int} + B_{shell^{int}} + \sum_{j=1}^{N} (B^j_{ext} + B^j_{shell^{ext}})$$

where B^i_{sens} is the magnetic field density measured by the i_{th} sensor, B_{int}, $B_{shell^{int}}$, B^j_{ext}, $B^j_{shell^{ext}}$ are the magnetic fields produced by the internal magnet, the surrounding steel spherical shell, the magnet of the j_{th} external connected FreeBOT and the steel spherical shell of the j_{th} FreeBOT. The internal magnet is static relative to magnetic sensors, so B_{int} is unchanged. The magnetic field produced by B^j_{ext} is blocked and guided into the steel spherical shell. This magnetic field distribution is heavily affected, which is visualized with the Ansys Maxwell simulator in Fig. 3. The magnetic field density reduces significantly after going through two steel shells, and the magnetic field direction is also changed. So, B^j_{ext} cannot be modeled by traditional magnet models.

The steel spherical shells are magnetized by magnets, and the sensors will perceive the magnetic field produced by the steel. $B^j_{shell^{ext}}$ can be divided into two parts: the steel around the external magnet that has been magnetized to saturation and the rest of the shell. The saturated part can be regarded as a permanent magnet and be modeled together with the external magnet. The remaining part can be ignored because the magnetic field decreases super-quadratic with distance, and this part is too away from the sensors. $B_{shell^{int}}$ can be approximated by the integral of infinitesimal steel elements:

$$B_{shell^{int}} = \int_{\phi=0}^{2\pi} \int_{\theta=0}^{pi} B_{steel}(m_{\phi,\theta}, r_{\phi,\theta,i}),$$

where $B_{steel}(m, r)$ is the magnetic field produced by an infinitesimal steel element at the magnetic sensor, m is the magnetic moment of the element, and r is the position of the sensor relative to the element. Such a magnetic field can be modeled as a magnetic dipole:

$$B(m, r) = \frac{\mu_0 m}{4\pi} \left(\frac{3r(\hat{m} \cdot r)}{||r||^5} - \frac{\hat{m}}{||r||^3} \right),$$

where μ_0 is the permeability of free space, and $\hat{m}$ is the normalized vector in the direction of m. However, the magnetic moment m of an arbitrary infinitesimal steel element depends on the magnetic field intensity produced by the magnets and hysteresis loop of steel. The hysteresis loop is nonlinear and history-dependent.

In a word, the steel spherical shell changes the magnetic field distribution of magnets. The magnetic field cannot be modeled by traditional magnet models. Therefore, we use the neural network to approximate f_Θ^{-1} and estimate the connection points.

C. Localization

Here, we propose a GCN-based algorithm to locate a variable number of connection points, which aims to locate multiple connection points with only single connection point training data.

In order to simplify the data collection, the connection point coordinates are discretized. As shown in Fig. 2, the upper hemisphere is split into 18 large regions, and each large region contains four sub-regions. When the internal magnet connects to some steel shell, the azimuth angle of connection points cannot be larger than $120°$. So, the lower hemisphere except for the region near the internal magnet is split into 12 regions. There is a magnetic sensor right below each large region. Due to the narrow space around two wheels, we do not place magnetic sensors near the wheels for now. So the regions at the lower hemisphere are not further split, and these regions contain only one sub-region. The upper hemisphere is much more frequently connected than the lower hemisphere. There are 84 uniform sub-regions in total when collecting data.

A pure multi-layer perceptron (MLP) can hardly learn to locate multiple connection points with only single connection point training data. A GCN-based localization algorithm is proposed, which achieves the multiple connection points localization by limiting the sensor receptive field. As shown in Fig. 4, 24 sensors are regarded as 24 nodes in an undirected graph. Six virtual nodes are added near the wheels. Two nodes are connected when two regions share one edge. For

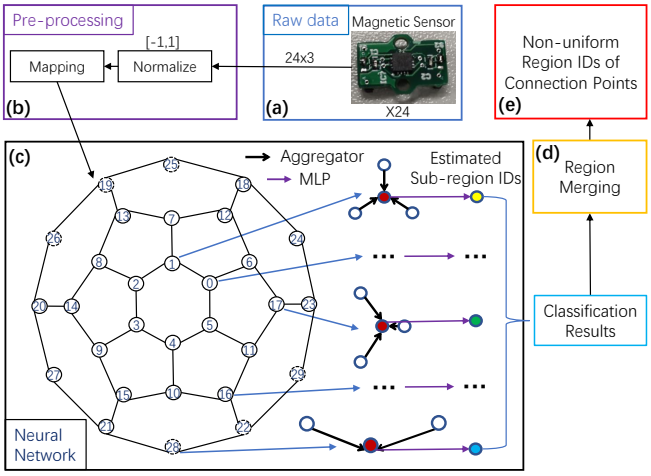

Fig. 4. Localization algorithm. (a) Raw data is fetched from the magnetic sensor array. (b) The data is stacked as a 24-by-3 matrix, which is then normalized to $[-1, 1]$. (c) A GCN-based network estimates sub-region classification results. (d) Region merging algorithm that merges classification results and estimates the non-uniform region ids of all connection points. (e) Final estimated non-uniform region ids. Full line circle: node (magnetic sensor); Dotted line circle: virtual node; Black arrow: aggregator; Purple arrow: MLP.

each node, an aggregator is used to aggregate the magnetic data from neighbors and generates an embedding. Then the embedding goes through an MLP, which estimates the labels. The classification labels contain M sub-region labels and one label that means no connection point, where M depends on the region division. In order to decrease the learning difficulty, the aggregators aggregate the nearest neighbors once. So, the estimation will not be influenced by remote sensors, which can hardly be well learned without a massive number of data. Also, the aggregators and MLPs are only shared for rotationally symmetric nodes so that the spatial distribution of sensors can be pre-coded into the network structure.

In this way, the nodes are divided into three groups for the upper hemisphere and six for the lower hemisphere. We can train nine groups of aggregators and MLPs to estimate the 30 regions, and each group can be trained separately. The MLP contains several fully connected layers and batch normalization layers so that the localization algorithm can run on our embedded controller in real-time.

Then we can get the final connection points position after merging the classification results of those 30 regions. This procedure is mainly based on the principle that two adjacent regions cannot be connected simultaneously. When two adjacent regions detect a connection point simultaneously, the connection point is near the edge of the two regions if sub-regions match. Otherwise, a predicted connection point is canceled based on prediction confidences. When more than two regions that share the same vertex detect a connection point simultaneously, the connection point is near the vertex. The sub-regions are also checked to guarantee better accuracy. In this way, the narrow areas around each edge and vertex can be regarded as new sub-regions, and 111 boundary sub-regions can be created. Finally, the whole steel spherical shell contains 199 non-uniform sub-regions. This procedure called region merging takes the set of estimated sub-region ids of the 30 large regions and outputs the non-uniform region ids of all connection points.

Algorithm 1 describes the region merging algorithm in more details, where R is the set of estimated the sub-region ids, M_{adj} is the sub-region adjacent matrix, HC is a value means high confidence, $GetConnectedSubGraphs$ calculates the connected sub-graphs of the sub-region graph, $GetConfidence$ gets the classification confidences of sub-regions in a graph, $CheckSubGraphConflict$ checks whether two graphs are too closed, $ShareVertex$ tells whether two graphs share a vertex, $Merge$ merges two graphs, $ShiftRegionAndMerge$ merges two graphs by shift two graphs closer, $RemoveGraph$ removes a graph (means a wrong classification), $UpdateGraph$ updates the data structure after changing the graphs, and $GraphSubRegionID$ gives the non-uniform region ids based on the regions in the graph.

Algorithm 1 RegionMerging(R)

$G = GetConnectedSubGraphs(R, M_{adj})$
$C = GetConfidence(G)$
$CT = CheckSubGraphConflict(G)$
while $len(CT) > 0$ **do**
  $i, j = GC[0]$
  if $ShareVertex(G[i], G[j])$ **then**
    $Merge(G[i], G[j])$
  else
    if $C[i] > HC$ **and** $C[j] > HC$ **then**
      $ShiftRegionAndMerge(G[i], G[j])$
    else if $C[i] > C[j]$) **then**
      $RemoveGraph(G[j])$
    else
      $RemoveGraph(G[i])$
    end if
  end if
  $UpdateGraph(G, CT)$
end while
return $GraphSubRegionID(G)$

The localization algorithm is summarized in Fig. 4.

III. EXPERIMENTAL RESULTS

Compared with the prototype of FreeBOT [9], more powerful processors and more sensors are added. The details can be found in Fig. 5.

A. Residual Magnetization Interference

Here, we test the residual magnetization magnitude caused by the steel spherical shell at different distances and compare it with the magnetic field of an external FreeBOT. We fixed a magnetic sensor at the center of the shell and also moved it near the shell. The steel shell is randomly magnetized. When the sensor is at the center of the shell, the magnetic field

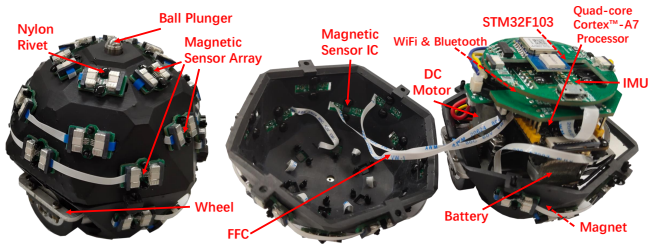

Fig. 5. FreeBOT equipped with new generation hardware. A FreeBOT is equipped with a Quad-core Cortex™-A7 processor, an STM32F103 processor, a magnetic sensor array, and an inertia measurement unit (IMU). The FreeBOT can communicate with other modules through WiFi and Bluetooth.

intensity is approximate 0.3 mT, which increases to 0.7 mT at maximum close to the shell. Then a FreeBOT is connected to the shell. The magnetic field intensity is approximate 0.8 mT at the opposite of the connection point, which increases to 3 mT at the center of the shell and 90 mT near the connection point.

The sensors are placed near the shell because the superposition of multiple connected FreeBOT magnetic fields can hardly be decomposed. The magnetic field measured far from the connection point has the same order of magnitude as residual magnetization. The residual magnetization is nonlinear and history-dependent, which would cause some uncertainty to the sensors. So we locate the connection points only with the sensors near the connection point, which can significantly reduce the residual magnetization interference.

B. Datasets

To train the neural network proposed in II-C, we need to collect the connection point coordinates related to the internal vehicle. Collecting such high-precision data might be complicated. Here, we use a simple way to collect the data with relatively low precision, which can meet our basic localization requirements. The region divide lines of the 84 sub-regions are manually painted on the steel spherical shells, which helps identify the sub-region id. Then the internal vehicle is placed in the lower hemisphere, and the region divide lines are aligned with the magnetic sensors manually. After aligning and fixing the upper hemisphere with the lower hemisphere, we can start collecting data. Another FreeBOT is connected to this FreeBOT and controlled carefully to move inside each sub-region in turn manually. Batches of data are collected with different order and magnetization history in this way. The training set and validation set are separately collected. Finally, the training set contains 50536 samples, and the validation set contains 21933 samples. Each sample contains the sub-region id and the magnetic data of the 24 sensors. Each region contains 2200 connected samples on average. The data order and label of datasets are further processed for each neural network before training. Due to the manual alignment error and the manual control error, the dataset is rough and may cause little region boundary shifting.

C. Training

The neural network is implemented with TensorFlow [23]. The connected data for a region can be the unconnected data for other regions, so the unconnected data percentage is more than 95% for each region, which is heavily imbalanced. So weighted cross-entropy loss is applied. The training hyperparameters are found heuristically: Adam as the optimizer, learning rate of 0.001, batch size of 128, and epochs of 30 with early stopping.

We need to train nine groups of aggregators and MLPs as mentioned in II-C. The training accuracy, validation accuracy, sub-region classification validation accuracy, and grouping information are shown in Table II. The sub-region accuracy represents the accuracy of estimating the sub-region id when there exists a connection point. The lower hemisphere has only one sub-region, so it has no sub-region accuracy. The included region id is the region id of large regions inside the group. Datasets contain some mislabeled samples caused by human error, which limits the training and validation accuracy. The boundaries of large regions and sub-regions are slightly shifted as training dataset distribution, which mainly decreases validation accuracy. With the region merging algorithm, the region boundary can be monitored by multiple networks, which can greatly improve the robustness for the multiple connection points localization.

TABLE II
TRAINING RESULTS

	Training Accuracy	Validation Accuracy	Sub-region Accuracy	Included Region ID
Group 1	92.2	91.8	85.2	[0-5]
Group 2	96.8	94.4	85.9	[6-11]
Group 3	96.1	95.3	88.4	[12-17]
Group 4	98.1	97.4	-	[18,21]
Group 5	97.4	94.6	-	[19,22]
Group 6	98.6	96.2	-	[20,23]
Group 7	99.2	97.1	-	[24,27]
Group 8	98.1	97.0	-	[25,28]
Group 9	98.5	97.5	-	[26,29]

D. Multiple Connection Points Localization

The trained model is transformed to TensorFlow TFLite models and then inference on the controller of FreeBOT. Finally, the algorithm can run at 40 Hz on FreeBOT.

In this experiment, three connection points localization is evaluated and demonstrated. Three FreeBOTs are connected to a FreeBOT equipped with magnetic sensors, and the three FreeBOTs are remotely controlled to change the system configuration. The connection point trajectories are manually observed and recorded, which are compared with localization results. As shown in Fig. 6, the regions are projected to a plane, and the region divide lines are distorted for intuitive visualization. Some observed connection point trajectories and the corresponding localization results of the three connected FreeBOTs are sampled and plotted on the plane with different colors. The localization results are discretely distributed on the region edges, vertices, and the center of

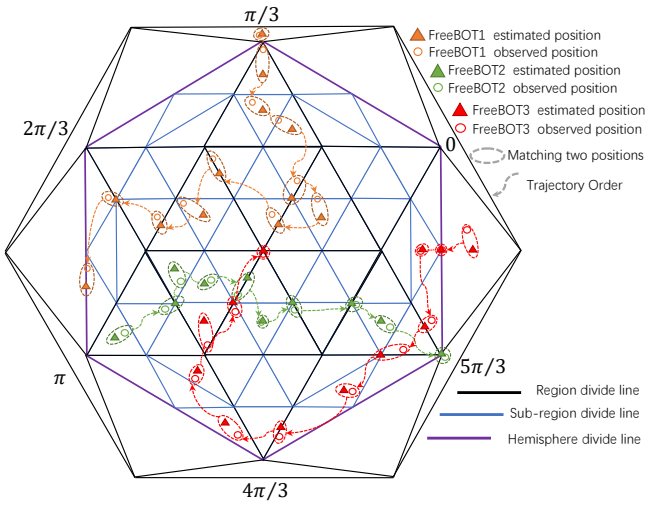

Fig. 6. Evaluation of multiple connection points localization performance. Black line: projected region divide line; Blue line: projected sub-region divide line; Purple line: divide line of the upper and lower hemisphere; Orange: the trajectory of FreeBOT1; Green: the trajectory of FreeBOT2; Red: the trajectory of FreeBOT3; Small triangle: discrete localization results; Small circle: observed connection point locations.

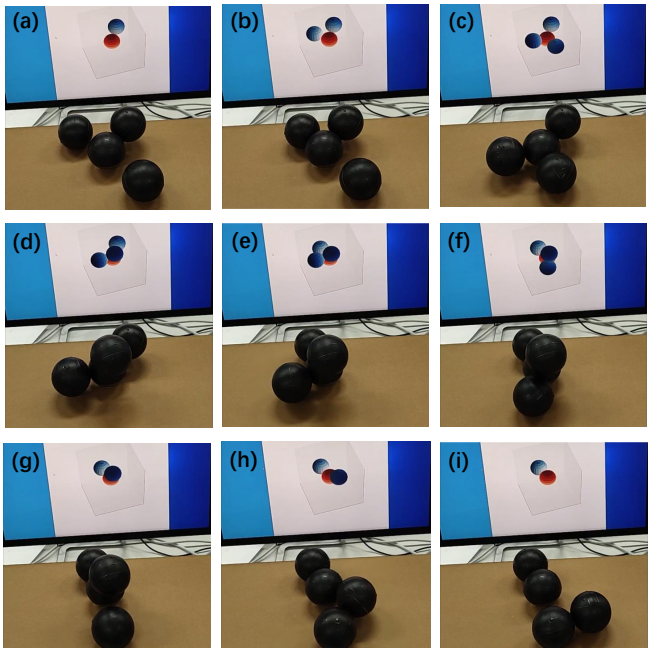

Fig. 7. Real-time configuration visualization. (a) Four FreeBOTs are placed on the table, and one FreeBOT has connected to the FreeBOT equipped with magnetic sensors. (b)-(c) The other two FreeBOTs connect to the central FreeBOT successively. (d)-(f) The FreeBOTs reconfigure around the central FreeBOT. (g)-(i) Two FreeBOTs disconnect successively. The central FreeBOT is fixed to stabilize the center of gravity. The magnet is at the whitest part of the visualized sphere on the screen.

sub-regions. For the 34 localization results, 76.5% of points are classified to the closest sub-region, while the other points are classified to the second closest sub-region. Assume that the sub-region position is at the center of it, and the mean value of the angles between estimated positions and observed positions to the center of the sphere is approximate $6.1°$, with the standard deviation of $3.1°$.

A real-time configuration display system is also implemented with Python and demonstrated. While the FreeBOTs reconfiguring, the localization results are transmitted back to a PC and visualized on the screen. Fig. 7 shows some captured pictures for the configuration visualization. Three FreeBOTs connect to a FreeBOT, reconfigure and then disconnect. The test subject and the visualization show good synchronization.

IV. DISCUSSIONS

To realize the obstacles crossing tasks proposed in [10], a minimum of 45 uniform regions classification is required. The 24 magnetic sensors achieve a good localization result in our experiments, fulfilling our basic localization precision requirement. The magnetic field decreases fast with the distance from the magnetic source. The angle between any two connection points is larger than $60°$. If a sensor is placed near the steel shell, it will not be influenced much by the steel and by the connected FreeBOT outside of the region over the sensor. This is the principle of the configuration detection system design, and it makes multiple connection point localization possible with only single connection point data. The 24 sensors are arranged near the steel shell based on this principle, and two adjacent regions cannot be connected simultaneously. This makes the final localization results more precise and robust. We can also place more magnetic sensors to achieve higher localization precision.

The proposed system can achieve good accuracy when a region is split into four sub-regions. However, the data is collected manually, and we can hardly collect higher precision data in this way. So, we do not test the limit accuracy of our system and systematically evaluate the localization accuracy here. The learned region division might also drift a little as training data distribution.

V. CONCLUSIONS AND FUTURE WORK

In this study, we proposed a localization system for FreeBOT configuration detection using a magnetic sensor array. A magnetic sensor array is designed for magnetic localization under ferromagnetic environments. After collecting enough data, the proposed system can localize a variable number of connection points in real-time accurately by applying a GCN-based algorithm. After synchronizing the localization systems in each FreeBOT, the whole FreeBOT system configuration can be detected, and the system can reconfigure automatically and accurately.

In future work, an automatic configuration data collection system will be introduced to further improve the system precision and better evaluate the system performance. More FreeBOTs will be equipped with a magnetic sensors array, and a distributed FreeBOT configuration detection system can be evaluated. Some other studies such as reconfiguration identification, localization fused with 6-axis IMU, and connection points tracking can also be considered.

REFERENCES

[1] S. Chennareddy, A. Agrawal, and A. K.R., "Modular Self-Reconfigurable Robotic Systems: A Survey on Hardware Architectures," *Journal of Robotics*, vol. 2017, pp. 1–19, Mar. 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] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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 *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

[10] H. Luo, M. Li, G. Liang, H. Qian, and T. L. Lam, "An obstaclescrossing strategy based on the fast self-reconfiguration for modular sphere robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

[11] 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.

[12] 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.

[13] J. Montero, M. Gherardini, F. Clemente, and C. Cipriani, "Comparison of online algorithms for the tracking of multiple magnetic targets in a myokinetic control interface*," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 2770–2776, iSSN: 2577-087X.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] Z. Sun, S. Foong, L. Maréchal, T. H. Teo, U.-X. Tan, and A. Shabbir, "Using heterogeneous sensory measurements in a compliant magnetic localization system for medical intervention," in *2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Jul. 2015, pp. 133–138, iSSN: 2159-6255.

[19] 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.

[20] 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.

[21] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs." in *ICML*, ser. JMLR Workshop and Conference Proceedings, M.-F. Balcan and K. Q. Weinberger, Eds., vol. 48. JMLR.org, 2016, pp. 2014–2023. [Online]. Available: http://dblp.uni-trier.de/db/conf/icml/icml2016.html#NiepertAK16

[22] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 1024–1034. [Online]. Available: http://papers.nips.cc/paper/6703-inductive-representation-learning-on-large-graphs.pdf

[23] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/