2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Dynamic Reconfiguration Integrated Nested A*: A Path Planner for Reconfigurable Robot to Improve Performance in Confined Spaces

W. K. R. Sachinthana, S. M. Bhagya P. Samarakoon, M. A. Viraj J. Muthugala, and Mohan Rajesh Elara

Abstract— **Path planning is crucial in numerous robotic applications. Reconfigurable robots possess the capability to alter their shape, enabling access to confined spaces, a task challenging for fixed-shape robots. However, existing path planners for reconfigurable robots are typically designed with predefined motion patterns for reconfiguration, lacking adaptation to space availability and executing reconfiguration only when the robot is static. This reliance on predefined patterns limits the potential of reconfigurable robots to navigate through confined spaces. This paper proposes a novel path-planning approach based on dynamic reconfiguration to address this limitation. The proposed method employs two nested A* algorithms modified to handle reconfiguration and efficient search, termed Dynamic Reconfiguration integrated Nested A* (DRiNA*). Experimental results demonstrate the proposed method's ability to find feasible paths for robot navigation using dynamic reconfigurations in confined spaces, surpassing the capabilities of existing path planners. The scalability of the proposed method to reconfigurable robots with varying numbers of blocks is also confirmed. Additionally, DRiNA* significantly reduces energy consumption compared to existing path planners.**

I. Introduction

Path planning is an essential feature in robotics, serving as the basis for navigating robots through diverse environments to achieve certain tasks [1], [2]. This involves the formulation and execution of paths for robots to traverse from an initial location to a target destination while considering various constraints and uncertainties [3], [4]. Numerous path-planning methods have been developed [1], [2]. For example, a cubic spline interpolation-based path planning method has been introduced in [5] to maintain the smoothness of moving the robot's path. This method uses a chaotic adaptive particle swarm optimization algorithm to optimize the path nodes in cubic spline interpolation. In path planning, the less computational time for optimizing the path is challenging. To overcome this problem, an efficient Q-learning algorithm that can find an optimal collision-free path within a short time has been proposed [6].

Sampling-based algorithms such as Rapidly Exploring Random Tree* (RRT*) could be often seen in path planning algorithms [7]. The work, [8], proposed PQ-RRT*, which combines potential functions based on RRT* and Quick-RRT* path planner to improve the efficiency of conventional RRT*. The work [9] has proposed a multi-objective dynamic RRT* composed of path generation and path replanning procedures to cope with dynamic environments. The A* algorithm is popular among path-planning algorithms [10]. Compared to RRT, the A* algorithm guarantees the finding of an optimal path if it exists and is preferred for low-dimensional problems. The modified A* planner proposed in [11] combines the jump point search algorithm, pruning algorithm, and quadratic bezier curve to reduce the calculation time, remove unnecessary nodes, and improve path smoothness. The work [12] has proposed an improved A* algorithm using a two-way optimization method with a cubic spline function to remove unnecessary nodes, shorten the path length, reduce the number of inflection points, and smooth the path. Most existing path planners discussed above have been developed for fixed-shape robots.

The accessibility of a fixed-shape robot is limited by geometrical constraints such as the narrowness of a passage [13]. Therefore, accessibility and navigation through confined spaces are challenging for fixed-shape robots. Reconfigurable robots have been introduced to alleviate this limitation in various application domains such as cleaning [14], exploration [15], search and rescue [16], and outer space [17]. The applications and challenges of reconfigurable robots are discussed in [18]. However, the path planning methods developed for fixed-shape robots (discussed earlier) cannot be directly applied to reconfigurable robots and require modifications to handle the reconfiguration ability [19].

Path-planning algorithms specially designed for reconfigurable robots can be found in the literature. In [20], the authors have developed a path planner for a reconfigurable robot using boustrophedon motion and a genetic algorithm. The path planner considered covering obstacles during navigation. An A* algorithm combined with a zigzag planner that generates waypoints to cover narrow spaces for a reconfigurable robot has been proposed in [21]. Energy efficiency is one of the crucial requirements for robots [22]. Therefore, attempts have also been made to develop energy-efficient path planners. For example, an energy-efficient and collision-free path planner for a reconfigurable robot in complex environments using Energy-efficient Batch-Informed Trees* (EBITR*) has been introduced in [23]. The work [24] proposed a path planner that combines heat conduction theory and grid-based optimization for a reconfigurable robot to plan efficient paths. Although path planners developed for reconfigurable robots could be found in the literature,

This research is supported by the National Robotics Programme under its National Robotics Programme (NRP) BAU, Ermine III: Deployable Reconfigurable Robots, Award No. M22NBK0054, A*STAR under its "RIE2025 IAF-PP Advanced ROS2-native Platform Technologies for Cross-sectorial Robotics Adoption (M21K1a0104)" programme, and also supported by SUTD Growth Plan (SGP) Grant, Grant Ref. No. PIE-SGP-DZ-2023-01.

W. K. R. Sachinthana, S. M. Bhagya P. Samarakoon, M. A. Viraj J. Muthugala, and Mohan Rajesh Elara are with the Engineering Product Development Pillar, Singapore University of Technology and Design, 8 Somapah Rd, Singapore 487372.

these path-planning methods consider predefined patterns for reconfiguration actions. Only the modules essential for reconfiguration are moved in these predefined patterns, while the other blocks are kept static. Furthermore, the reconfiguration pattern is fixed and not adaptable per the space availability. Therefore, the performance of existing path-planning methods for reconfigurable robots is limited in confined spaces, thereby restricting the true potential of reconfigurable robots.

This paper proposes a novel path-planning approach using the dynamic reconfigurability of a reconfigurable robot to navigate through confined spaces. Dynamic reconfigurability in the proposed approach enables a robot to adapt to space requirements, facilitating space-efficient maneuvering and paving the way for feasible accessibility and navigation in confined spaces. Based on the existing literature, the path-planning strategy proposed in this paper represents the first attempt to utilize the dynamic reconfigurability of a reconfigurable robot for path planning and navigation. The reconfigurable robot model considered for this work is explained in Section II. Section III details the proposed path planning algorithm with the reasoning behind dynamic reconfiguration. Particulars on the validation and performance comparison of the proposed algorithm are discussed in Section IV. Conclusions of the work are given in Section V.

II. The Reconfigurable Robot Model

Smorphi[1] is a self-reconfigurable robot with a series of square-shaped blocks connected through hinges. Each module of the robot is equipped with a mecanum wheel-driven locomotion unit, which allows omnidirectional locomotion. This mechanism supports the relative motions between individual robot modules, where the robot can perform shape-shifting by rotating modules around the hinges. The number of modules in a robot can be varied by attaching modules to either end of the robot through hinges, which increases the possible morphologies that the robot can achieve. A robot platform with four modules is depicted in Fig. 1.

The kinematics of the robot is essential for defining the positions of each module of the robot, given the input parameter values. As the robot allows both linear and rotational movements of each module, the kinematic model should be created to derive both linear and angular positions of each module. A reconfigurable robot with n number of modules as depicted in Fig. 2 is considered for deriving the kinematic model. The first module of the robot is considered the reference where MP_1: (X_0, Y_0) is the center position. It is considered that the i^{th} module is rotated by q_i with respect to $(i-1)^{\text{th}}$ module due to hinge reconfiguration for all $i = 1$ to n. The model resembles a serially connected chain structure in which the base joint moves. Thus, any specific position of the robot with respect to the world frame can be derived when the reference and the angular position parameters (i.e., $q_i, i = 1$ to n) are known. The set of angular posistions are defined by Q such that $Q = [q_1, q_2, .., q_n]$. The corners of each module are derived here to check the collision with obstacles during the dynamic reconfiguration.

The position of the serial link chain node where the hinge that connects the $(i-1)^{\text{th}}$ block with the i^{th} block is considered CP_i for all $i = 2$ to n. Let the lengths of the serial link between the i^{th} hinge and the $(i+1)^{\text{th}}$ hinge be l_i for $i = 1$ to $n-1$. The position offset of the $(i+1)^{\text{th}}$ module, ΔP_i with respect to the reference module corner position CP_1 can be obtained as in (1) for $i = 2$ to n. Then hinge positions CP_i, $i = 2$ to n can be calculated as in (2).

$$[\Delta p_i]_{1\times 2} = l_{i-1}\left[\cos(\textstyle\sum_{r=1}^{i-1} q_r) \quad \sin(\sum_{r=1}^{i-1} q_r)\right]_{1\times 2} \tag{1}$$

$$[CP_i]_{1\times 2} = \left[CP_1\right]_{1\times 2} + \left[\textstyle\sum_{r=1}^{i-1} \Delta p_r\right]_{1\times 2} \tag{2}$$

Knowing the hinge positions and the angular orientations, all the other corner positions of the i^{th} module can be calculated. The other corners of the i^{th} are defined as CP_i^1, CP_i^2, and CP_i^3 (refer Fig.2). Three scenarios can be considered given the reference hinge position of i^{th} module reference position (i.e., CP_i) and the reference hinge position for $(i+1)^{\text{th}}$ module reference position (i.e., CP_{i+1}). The positions CP_i^1, CP_i^2, CP_i^3 can be calculated as in (3), where w_i is given in (4) per the scenario and L is the width of a module. Here C and S denotes sin and cos, respectively.

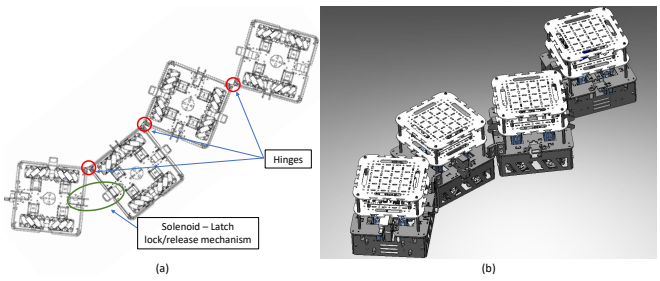

Fig. 1. Reconfigurable robot with four module hardware configuration (a). Bottom view (b). Isometric view.

[1]www.wefaarobotics.com

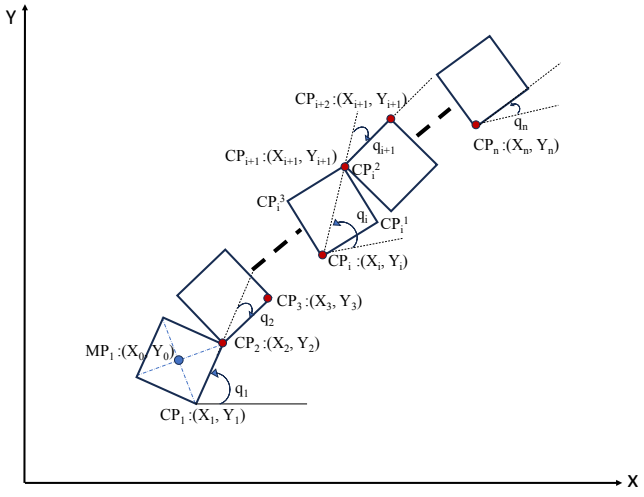

Fig. 2. Kinematics Model of the Reconfigurable Robot

$$\begin{bmatrix} CP_i^1{}_{(1\times2)} \\ CP_i^2{}_{(1\times2)} \\ CP_i^3{}_{(1\times2)} \end{bmatrix} = \begin{bmatrix} CP_{i(1\times2)} \\ CP_{i(1\times2)} \\ CP_{i(1\times2)} \end{bmatrix} + L \begin{bmatrix} \mathrm{C}(w_i) & \mathrm{S}(w_i) \\ \mathrm{C}(w_i) & \mathrm{S}(w_i) \\ \mathrm{C}(w_i) & \mathrm{C}(w_i)+\mathrm{S}(w_i) \\ \mathrm{C}(w_i) & \mathrm{S}(w_i) \end{bmatrix} \tag{3}$$

$$w_i = \begin{cases} q_i & \text{if serial link is between } CP_i \text{ and } CP_i^1 \\ q_i - 45 & \text{if serial link is between } CP_i \text{ and } CP_i^2 \\ q_i - 90 & \text{if serial link is between } CP_i \text{ and } CP_i^3 \end{cases} \tag{4}$$

III. Path Planning

A. Importance of Dynamic Reconfiguration

The existing path planning methods of reconfigurable robots (e.g. [21], [24], [23]) use fixed motion patterns to perform reconfigurations. When performing a reconfiguration, only a few modules that change the configuration are moved at a time, while the rest of the modules stay static until the reconfiguration is completed. Fig. 3(a) explains the requirement of free space (indicated in grey color) to perform shape transition from 'O' shape ($Q = [90°, 0°, -135°, -135°]$) to 'I' shape ($Q = [180°, 0°, 45°, -135°]$) of a four module robot using a predefined pattern for reconfiguration. If the space is not sufficient, these existing approaches fail to produce a feasible path (see Fig.3(b)). Therefore, the usability of the existing path-planning methods of reconfigurable robots is limited in confined spaces. If reconfiguration is performed while the robot is moving (see Fig.3(c)), it can overcome this limitation. As the robot moves linearly while performing the module rotations, it can use the narrow space to avoid space constraints to do shape transitions. The way of reconfiguration should not be predefined here to cater to the space requirements. The execution of reconfigurations while moving the robot's blocks in linear and rotation movements within their limits as required for space constraints without relying on a predefined pattern is considered dynamic reconfiguration in this paper. Therefore, reconfigurations may utilize distinct motion patterns of each module depending on the space available for reconfiguration. Furthermore, dynamic reconfiguration uses simultaneous rotational and linear movement of each module within its limits, allowing the robot to use the available space more efficiently. The consideration of dynamic reconfigurations could improve the feasibility of path planning in confined spaces.

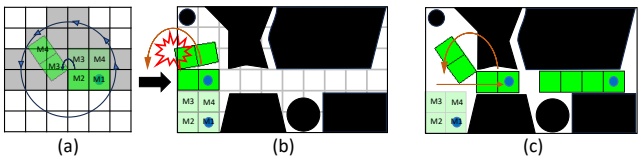

Fig. 3. (a) and (b): Limitation of the existing path planning of reconfigurable robots that use predefined pattern for reconfiguration (i.e.,[21], [24], [23]), and (c): How dynamic reconfigurations could overcome the limitation.

B. Dynamic Reconfigurability integrated Nested A* (DRiNA*)

The A* algorithm can be considered one of the best path-planning algorithms that can be used for metric environments. This algorithm consists of the strengths of Dijkstra's algorithm and greedy best-first search [10]. By combining heuristic and shortest-path searching, it can explore the search space efficiently. Here, initially, the environment is transformed into a grid cell map, which helps create a graph-based search methodology. The nodes indicate each grid cell, where the edges represent the connection between cells. The key parameters of the A* algorithm are the cost g, which represents the actual cost from the start to the current node, the heuristic component h, which is an estimated cost value from the current node to the goal node, and the total cost f which is the sum of above two parameter values. The proposed system uses an A* algorithmic approach to determine the shortest path between the start and the end node. Using the traditional approach is enough for a fixed-shaped robot to derive the path. However, as the proposed system utilizes a reconfigurable robot, it is necessary to alter the algorithm parameters to consider the reconfigurability. Therefore, start and goal nodes are represented by the combination of the linear position coordinate of the reference single module and the angular positions of all the modules. Each step of the algorithm lets the robot move each module freely within the given constraints. The grid cell map consists of smaller cells of one unit equal to 1cm x 1cm. At each step, the robot is allowed to linearly move 1 unit cell in four directions, which are up, down, left, and right, and a rotational movement of $5°$ clockwise or anticlockwise. So, considering all the permutations of the robot's movement per step, given the number of modules n, the number of permutations can be calculated as in (5). At a considered step of A*, each node produces children nodes equivalent to the number of possible permutations in (5).

$$permutations = 4 * 3^n \tag{5}$$

Critical parameters of the A* algorithm for the reconfigurable robot platform are straightforward. As for the cost value g, it calculates the total movement of the robot modules between the current node and the starting node. Let the current node linear and angular positions are $[(x_c, y_c), (q_{c_1}, q_{c_2}, ..)]$ and for the child node $[(x_{ch}, x_{ch}), (q_{ch_1}, q_{ch_2}, ...)]$. Let L be the side length of the robot. Then, the cost g of the child node can be expressed as in (6).

$$child.g = (x_{ch} - x_c) + (y_{ch} - y_c) + L * (|q_{ch_1} - q_{c_1}| + |q_{ch_2} - q_{c_2}| + ...) \tag{6}$$

To calculate the heuristic estimate to the goal position, Manhattan distance in between particular child node and goal node linear positions of the reference single module is used with the angular difference between the child configuration

and the goal configuration. Let the child node position be $[(x_{ch}, y_{ch}), (q_{ch_1}, q_{ch_2}, ...)]$ and the goal node position be $[(x_g, y_g), (q_{g_1}, q_{g_2}, ...)]$. Then, the heuristic component can be calculated as in (7).

$$child.h = |x_g - x_{ch}| + |y_g - y_{ch}| + L*(|q_{g_1} - q_{ch_1}| + |q_{g_2} - q_{ch_2}| + ...) \quad (7)$$

Considering the number of children nodes created by a single parent node, it takes a significant amount of time to provide a solution for a given environment. Therefore, changes have been applied to the path planning algorithm to reduce excess time, considering the nature of the reconfigurable robot. For the robot to move through a particular path using reconfiguration, the path should be wide enough to fit at least a single module of the robot. Considering this fact, a new heuristic component is introduced instead of Manhattan distance. Here, a separate A* is applied only to the reference single module to derive the shortest path between the current node and the goal node. This A* algorithm outputs a feasible path that only a single module of the robot can follow, and it takes less computational time. The length of this derived path is then used as the heuristic component for linear position difference. Thus, the updated heuristic component for outer A* can be calculated as in (8).

$$child.h = A^*_{Inner}((x_{ch}, y_{ch}), (x_g, y_g)) + L*(|q_{g_1} - q_{ch_1}| + ...) \quad (8)$$

Calculating parametric components for the inner A* is straightforward as it considers only a single module of the robot. Cost g value is calculated considering the current linear position and the last position of the robot. Cost h is derived using the Manhattan distance between the current and end positions. That way, the modified algorithm avoids creating unnecessary nodes and saves time. Additionally, the linear distance heuristic component improves path quality by diminishing the impact of the cost g, resulting in the reduction of more unnecessary nodes within the search space. In brief, the outer A* handles the angular orientation variations while the inner A* handles the linear position variations. Outer A* mainly considers the effect of the heuristic component, so it almost resembles the behavior of a greedy best-fit search algorithm. The overall flow of the proposed algorithm is explained in Algorithm 1.

Algorithm 1 DRiNA* path planning

Input $[MP_{start}, Q_{start}, MP_{goal}, Q_{goal}]$
Output *robot path*

```
map ← create grid map;
Node ← [MP, Q : [q1, q2, ...], Node.parent, g, h, f]
Node_start ← Node(MP_start, Q_start, None, 0, 0, 0)
Node_goal ← Node(MP_goal, Q_goal, None, 0, 0, 0)
openlist, closedlist
openlist.insert(Node_start)
move_linear(5×2) ← [[0, 0], [0, −1], [−1, 0], [0, 1], [1, 0]]
move_ang(81×4) ← [[0, 0, 0, 0], [5, 0, 0, 0], [−5, 0, 0, 0], ...]
while !openlist empty do
    Node_cur ← Select node with lowest f value
    if Node_cur == Node_goal then
        return path
        break
    end if
    for linear step in move_linear do
        h_l = Inner A*(Node_new.MP, Node_new.MP)
        for angular step in move_ang do
            Node_new.MP = Node_cur.MP + linear step
            Node_new.ang = Node_cur.ang + angular step
            Node_new.h = h_l
            if Node_new out of bounds | collision then
                continue
                children.append(Node_new)
            end if
        end for
    end for
    for child in children do
        set g, h, f scores as in (6) and (7)
    end for
end while
```

C. Obstacle Collision Detection

Obstacle avoidance is also a part of the A* algorithm which ensures to create children nodes that are not colliding with the obstacles in the environment and inside the boundaries. As each robot module is free to move in the proposed system, it is necessary avoid self collisions as well. These conditions have been considered in the proposed system in order to create a suitable collision detection system. Here, the robot itself and the all the obstacles are represented as polygons which can contain multiple vertices connected through edges. In order to check for the collision, intersection between polygons are considered. Obstacles are inflated by 2 cm to maintain tolerance. Let the robot consists of modules $M1, M2, M3, ..$ with each consists of polygons, $Poly_1 : [CP_1^1, CP_1^2, CP_1^3, CP_1^4], Poly_2 : [CP_2^1, CP_2^2, CP_2^3, CP_2^4], ..$ each represented by the corner positions of particular module. The obstacles be, $Obs_1 : [O_1^1, O_1^2, O_1^3, O_1^4, ...], Obs_2 : [O_2^1, O_2^2, O_2^3, O_2^4, ...], ...$ Then algorithm checks for the intersections in between considered edges of the polygons. If there's such an intersection, algorithm identifies it as a collision. But if the intersection point would be either corner of the edges, those scenarios are neglected.

IV. Results and Discussion

A. Experimental Setup

The proposed path planning method has been validated by conducting several simulations utilizing a virtual model of Smorphi reconfigurable robot. The built virtual model follows the exact dimensions and kinematics constraints of the actual robot. The simulations were carried out in a Python-based simulation environment. Even though a step

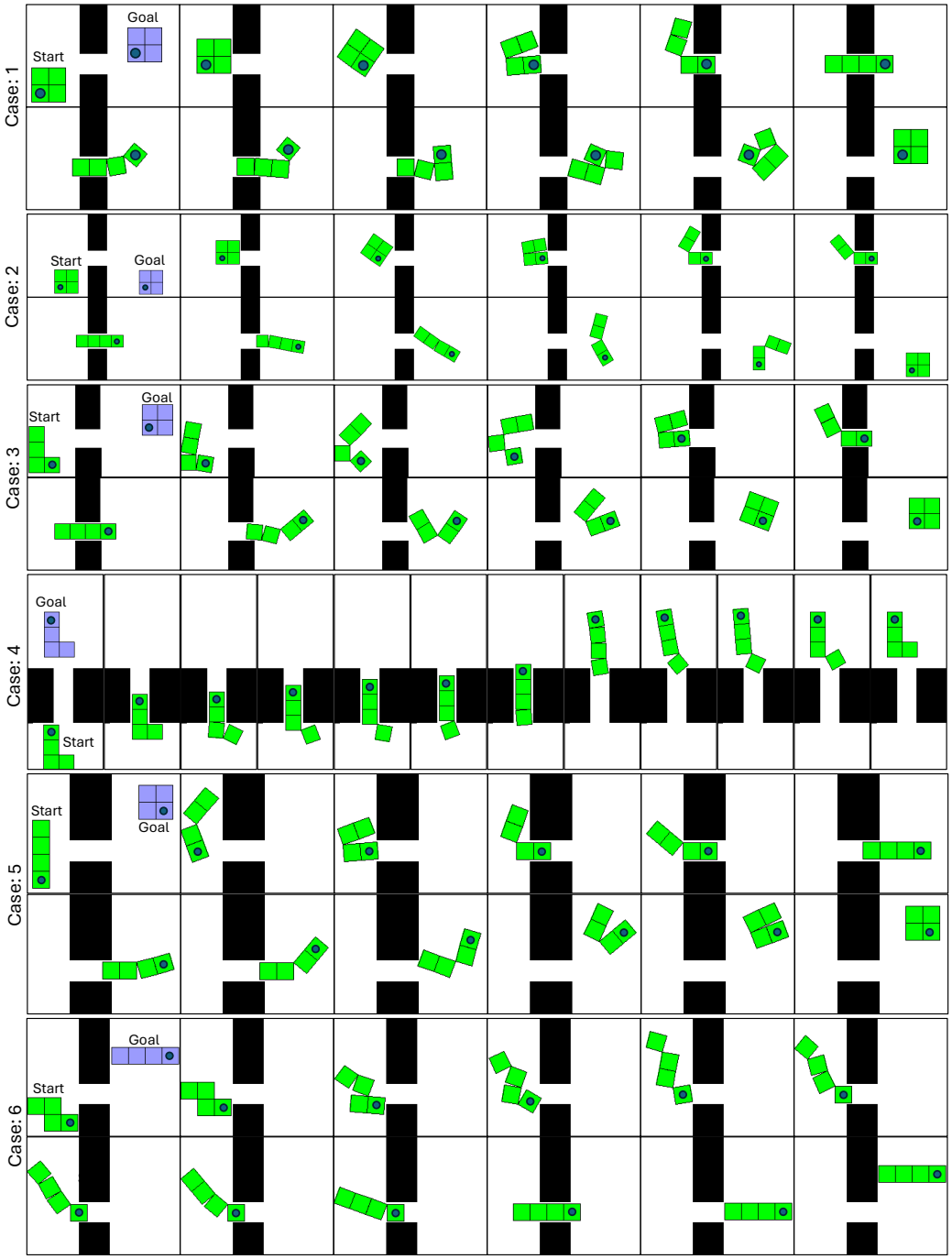

Fig. 4. Experiment results for four module configuration used for performance comparison.

movement of the robot has been represented using discrete variables (MP, Q), resultant positions (corner positions) of the all-robot modules are calculated as floating-point values to preserve the accuracy. Also, each module of the robot is allowed to perform rotational movements. Therefore, the whole environment has been represented as a continuous space where robots and obstacles are represented as polygons. Two sets of experiments were conducted. The first set considered the comparison of the performance of the proposed method against the existing path-planning methods of reconfigurable robots. The second experiment focuses on the scalability of the proposed method for a reconfigurable robot with a different number of blocks.

B. Performance evaluation

The existing path planning methods of reconfigurable robots, the modified A* proposed in [21], heat conduction-based planning proposed in [24], and EBITR* proposed in [23] have been considered for comparing the performance of the proposed DRiNA*. A reconfigurable robot with four modules has been considered here since the existing method is only developed considering four blocks and robots. Six test

cases shown in Fig. 4 have been considered for the comparison. The ability of reconfigurability in the given environment and the distance cost defined by the total distance traveled by all the robot modules have been considered the performance matrix. The travelled distance is also chosen as metric since the enegy usage is a concern for mobile robots. The obtained results are given in Table I.

In test case 1 given in Fig. 4(a), the goal of the planning was to move the robot in the initial position with the 'O' ($Q = [90°, 0°, -135°, -135°]$) to the given goal location with the goal configuration of 'O' shape. Here, the robot must navigate through a narrow passage by reconfiguring itself. With the proposed DRiNA* approach, the robot successfully navigated and reached the goal position with the desired goal configuration. According to the working principle of the existing approaches, the robot must first reconfigure itself to 'I' shape ($Q = [180°, 0°, 45°, -135°]$) to move through the narrow space and then reconfigure itself to 'O' after passing the narrow passage. However, the space on the left side is insufficient for facilitating this reconfiguration since those methods used pre-defined patterns for reconfiguration, as explained in Section III-A. Therefore, all the considered existing methods failed to produce a feasible path with reconfigurations for the robot to follow to reach the goal with the desired shape configuration.

In case 2, all the algorithms were able to provide a solution to find a path and reconfigurations to reach the goal with the given shape configuration. The navigation task in this case was similar to case 1, where the initial and goal configurations are 'O' while the robot needs to move through a narrow passage. In contrast, the space availability on both sides of the narrow pass is much higher compared to case 1. This higher space availability facilitated the robot reconfiguration into the 'I' shape first, then navigating through the narrow passage, and finally reconfiguring back to the 'O' shape at the goal location in the events of the existing methods. However, the traveled distances observed for the existing methods are way higher than that of the proposed method in this case. The traveled distance can be used as a metric to estimate the energy usage of the robot. From that perspective, the existing methods, EBITR*, heat conduction, and the modified A*, have recorded 7.14%, 16.95%, and 16.95% more energy consumption than the proposed DRiNA*.

A similar behavior was observed in case 3, where the robot in its initial 'L' configuration ($Q : [90°, 0°, 45°, 135°]$) had to reach the goal with the 'L' shape. However, the modified A* proposed in [21] was unsuccessful in navigating the robot to the goal. The least distance cost was observed from the proposed DRiNA* method in case 3, too. In cases 4, 5, and 6, the proposed DRiNA was successful in navigating the robot as demanded. In contrast, several existing methods have only been able to navigate the robot in case 4 but case 5,6 failed.

According to these results, the proposed method surpassed

TABLE I
RESULTS COMPARISON

	Method	Possibility of navigation	Navigation cost
Case1	Modified A*	No	*
	Heat conduction	No	*
	EBITR*	No	*
	DRiNA* (proposed)	Yes	1543.207
Case2	Modified A*	Yes	2083.389
	Heat conduction	Yes	2083.389
	EBITR*	Yes	1908.504
	DRiNA* (proposed)	Yes	1781.302
Case3	Modified A*	No	*
	Heat conduction	No	*
	EBITR*	No	*
	DRiNA* (proposed)	Yes	1619.541
Case4	Modified A*	No	*
	Heat conduction	Yes	1209.515
	EBITR*	Yes	1209.515
	DRiNA* (proposed)	Yes	1102.755
Case5	Modified A*	No	-
	Heat conduction	No	*
	EBITR*	No	*
	DRiNA* (proposed)	Yes	1656.76
Case6	Modified A*	No	*
	Heat conduction	No	*
	EBITR*	No	*
	DRiNA*(proposed)	Yes	1244.306

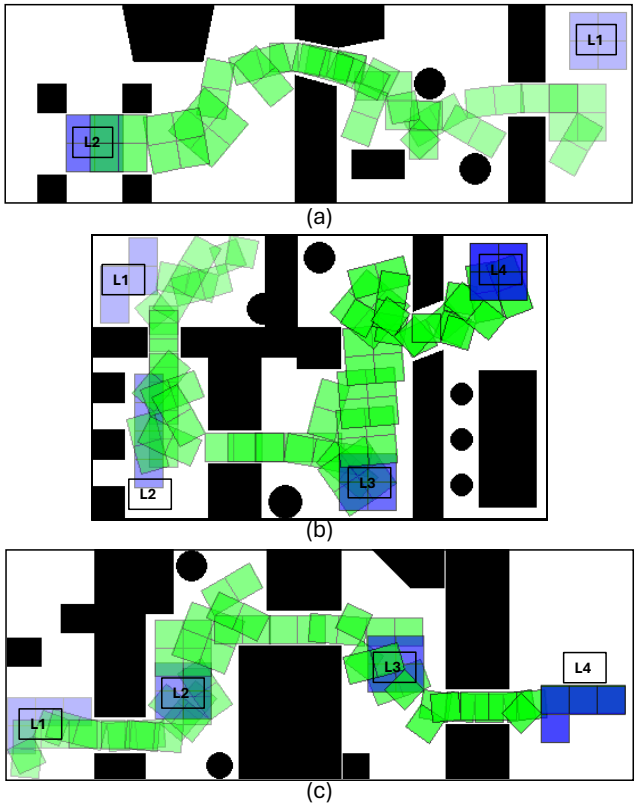

Fig. 5. DRiNA* path planning for complex environments. (a).The robot initially at L1 in 'O' configuration ($Q = [90°, 0°, 135°, 135°]$) and commanded to moved to L2 with 'O' configuration ($Q = [90°, 0°, 135°, 135°]$) as the end. (b) The robot initially at L1 in 'Z' configuration ($Q = [180°, 180°, 45°, 45°]$) and commanded to move to L2 with 'I' configuration ($Q = [90°, 0°, 45°, 135°]$), then to L3, with 'O' ($Q = [90°, 0°, 135°, 135°]$), then finally L4 with 'O' ($Q = [270°, 0°, 135°, 135°]$) as the end configuration. (C). The robot initially at L1 in 'L' ($Q = [180°, 0°, 45°, 45°]$) configuration and commanded to move to L2 with 'O' configuration ($Q = [270°, 0°, 135°, 135°]$), then L3 with 'O' ($Q = [180°, 0°, 135°, 135°]$), then finally L4 with 'L' ($Q = [180°, 0°, 45°, 45°]$) as the end configuration.

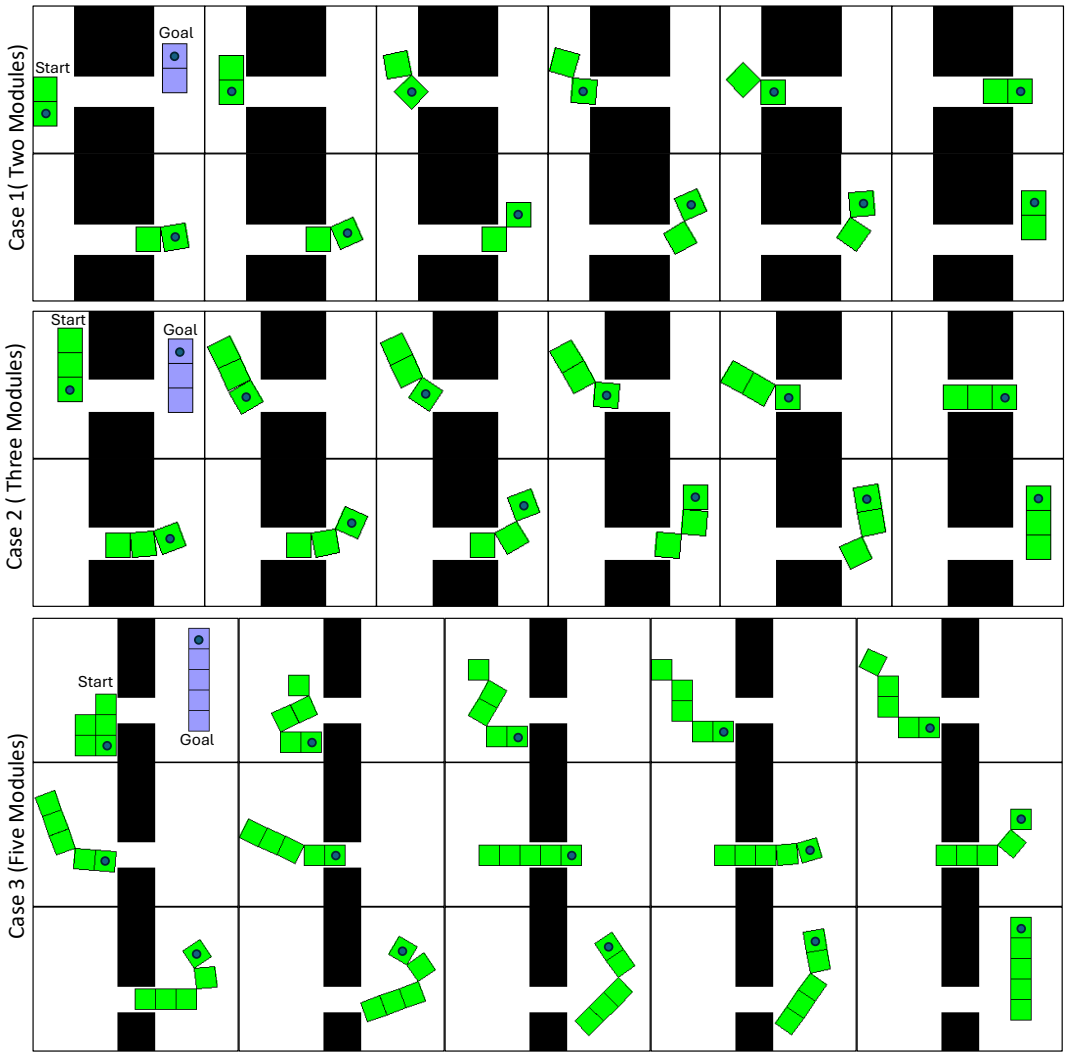

Fig. 6. Experiment results for two, three, five module configuration

the performance of the existing path planning methods in all the above cases due to the use of space-aware dynamic reconfiguration. Even though minimizing energy consumption is not a priority of the proposed DRiNA* algorithm, it has been able to surpass existing methods in this aspects, demonstrating the growing potential. Apart from these six cases, situations of a robot navigating through complex environments with many confined spaces as shown in Fig. 5 were also considered for demonstrating the full potential of the proposed method. In the scenarios shown in Fig. 5(a), initially the robot in 'O' configuration was at the location 'L1' and it was commanded to move to the location 'L2' where the end configuration should be 'O' configuration. The scenarios shown in Fig. 5(b) and Fig. 5(c), the robot was commanded with two intermediate way points 'L2' and 'L3' with different shape reconfiguration requests. As similar to the previous case, the proposed method was capable of navigating the robot toward the goals and achieving the desired configuration at the goals. In contrast, the existing path planners of reconfigurable robots could not successfully achieve this kind of navigation behavior. Therefore, the proposed DRiNA* can bring out the full potential of a reconfigurable robot with the help of dynamic reconfiguration.

C. Scalability Analysis

The second set of simulations have been carried out to demonstrate the scalability of the proposed method with reconfigurable robot with different number of blocks. Two, three and five module configurations of the robot have been considered here. The robot was assigned a navigation task for each case similar to the first set of experiments. Different confined environments were considered here to facilitate the size variation of the robot due to number of blocks. The traced paths and the configuration of the robots are shown in Fig. 6. In all the test cases, the robots were capable of reaching the goal with the desired configuration using the proposed DRiNA* method demonstrating its scalability. The existing path planning methods (A*, EBITR*, Heat conduction) are developed to work only with robots with four

blocks and the scalability for robots with different number of blocks is not possible. Therefore, the scalability is an added advantage of the proposed DRiNA* compared to the existing path planning methods.

V. Conclusion

This paper proposed a novel path-planning approach for reconfigurable robots to enhance performance in confined spaces. The proposed method leverages the dynamic reconfiguration capability of robots to plan an efficient path and adapt reconfiguration based on space availability. The path planner is developed using two nested A* algorithms, which have been modified to accommodate dynamic reconfiguration while enhancing search efficiency.

The proposed Dynamic Reconfigurability integrated Nested A* (DRiNA*) has been compared against existing path planners for reconfigurable robots. According to the experimental results, the proposed DRiNA is capable of generating feasible navigation paths using dynamic reconfiguration in confined spaces, where existing path planners have failed. Furthermore, the proposed DRiNA demonstrates superior energy efficiency compared to existing path planners. Additionally, DRiNA exhibits scalability to reconfigurable robots with varying numbers of blocks, addressing a limitation of existing path planners. Future work includes extending the method to handle dynamic obstacles and unknown environments.

References

[1] M. N. Ab Wahab, S. Nefti-Meziani, and A. Atyabi, "A comparative review on mobile robot path planning: Classical or meta-heuristic methods?" *Annual Reviews in Control*, vol. 50, pp. 233–252, 2020.

[2] B. Patle, A. Pandey, D. Parhi, A. Jagadeesh *et al.*, "A review: On path planning strategies for navigation of mobile robot," *Defence Technology*, vol. 15, no. 4, pp. 582–606, 2019.

[3] B. Sotolongo, A. Dutta, S. Sisley, and G. Sharma, "Shortest path planning with an energy-constrained robot," in *2021 ieee international conference on systems, man, and cybernetics (SMC)*. IEEE, 2021, pp. 745–750.

[4] X. Deng, R. Li, L. Zhao, K. Wang, and X. Gui, "Multi-obstacle path planning and optimization for mobile robot," *Expert Systems with Applications*, vol. 183, p. 115445, 2021.

[5] J. Lian, W. Yu, K. Xiao, and W. Liu, "Cubic spline interpolation-based robot path planning using a chaotic adaptive particle swarm optimization algorithm," *Mathematical problems in engineering*, vol. 2020, pp. 1–20, 2020.

[6] A. Maoudj and A. Hentout, "Optimal path planning approach based on q-learning algorithm for mobile robots," *Applied Soft Computing*, vol. 97, p. 106796, 2020.

[7] L. G. D. Véras, F. L. Medeiros, and L. N. Guimaráes, "Systematic literature review of sampling process in rapidly-exploring random trees," *IEEE Access*, vol. 7, pp. 50 933–50 953, 2019.

[8] Y. Li, W. Wei, Y. Gao, D. Wang, and Z. Fan, "Pq-rrt*: An improved path planning algorithm for mobile robots," *Expert systems with applications*, vol. 152, p. 113425, 2020.

[9] J. Qi, H. Yang, and H. Sun, "Mod-rrt*: A sampling-based algorithm for robot path planning in dynamic environment," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 8, pp. 7244–7251, 2020.

[10] A. Candra, M. A. Budiman, and K. Hartanto, "Dijkstra's and a-star in finding the shortest path: A tutorial," in *2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA)*. IEEE, 2020, pp. 28–32.

[11] X. Wang, Z. Liu, and J. Liu, "Mobile robot path planning based on an improved a* algorithm," in *International Conference on Computer Graphics, Artificial Intelligence, and Data Processing (ICCAID 2022)*, vol. 12604. SPIE, 2023, pp. 1093–1098.

[12] L. Zhang and Y. Li, "Mobile robot path planning algorithm based on improved a star," in *Journal of Physics: Conference Series*, vol. 1848, no. 1. IOP Publishing, 2021, p. 012013.

[13] M. A. V. J. Muthugala, S. M. B. P. Samarakoon, and M. R. Elara, "Design by robot: A human-robot collaborative framework for improving productivity of a floor cleaning robot," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 7444–7450.

[14] S. M. B. P. Samarakoon, M. A. V. J. Muthugala, A. V. Le, and M. R. Elara, "Htetro-infi: A reconfigurable floor cleaning robot with infinite morphologies," *IEEE Access*, vol. 8, pp. 69 816–69 828, 2020.

[15] C. Liu and M. Yim, "Reconfiguration motion planning for variable topology truss," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1941–1948.

[16] L. Yehezkel, S. Berman, and D. Zarrouk, "Overcoming obstacles with a reconfigurable robot using reinforcement learning," *IEEE Access*, vol. 8, pp. 217 541–217 553, 2020.

[17] W. Reid, R. Fitch, A. H. Göktoğan, and S. Sukkarieh, "Sampling-based hierarchical motion planning for a reconfigurable wheel-on-leg planetary analogue exploration rover," *Journal of Field Robotics*, vol. 37, no. 5, pp. 786–811, 2020.

[18] M. Yim, W.-M. 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.

[19] W. Cheah, T. B. Garcia-Nathan, K. Groves, S. Watson, and B. Lennox, "Path planning for a reconfigurable robot in extreme environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 10 087–10 092.

[20] S. B. P. Samarakoon, M. V. J. Muthugala, and M. R. Elara, "Global and local area coverage path planner for a reconfigurable robot," in *2022 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2022, pp. 1–8.

[21] A. V. Le, V. Prabakaran, V. Sivanantham, and R. E. Mohan, "Modified a-star algorithm for efficient coverage path planning in tetris inspired self-reconfigurable robot with integrated laser sensor," *Sensors (Basel, Switzerland)*, vol. 18, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:51934967

[22] M. A. V. J. Muthugala, S. M. B. P. Samarakoon, and M. R. Elara, "Tradeoff between area coverage and energy usage of a self-reconfigurable floor cleaning robot based on user preference," *IEEE Access*, vol. 8, pp. 76 267–76 275, 2020.

[23] P. T. Kyaw, A. V. Le, P. Veerajagadheswar, M. R. Elara, T. T. Thu, N. H. K. Nhan, P. Van Duc, and M. B. Vu, "Energy-efficient path planning of reconfigurable robots in complex environments," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2481–2494, 2022.

[24] H. Do, L. Anh Vu, O. Weeger, R. E. Mohan, Y. Guo, N. Tan, M. Vu, and V.-D. Phan, "Path planning for reconfigurable htetro robot combining heat conduction-based and discrete optimization," *IEEE Access*, vol. PP, pp. 1–1, 09 2021.