mathematics

MDPI

Article

Inter-Reconfigurable Robot Path Planner for Double-Pass Complete Coverage Problem

Ash Wan Yaw Sang [1], Zhenyuan Yang [1], Lim Yi [1], Chee Gen Moo [1], Rajesh Elara Mohan [1], Anh Vu Le [2,*]

[1] ROAR Lab, Engineering Product Development, Singapore University of Technology and Design, Singapore 487372, Singapore; ash_wan@mymail.sutd.edu.sg (A.W.Y.S.); zhenyuan_yang@mymail.sutd.edu.sg (Z.Y.); yi_lim@mymail.sutd.edu.sg (L.Y.); cheegen_moo@sutd.edu.sg (C.G.M.); rajeshelara@sutd.edu.sg (R.E.M.)

[2] Communication and Signal Processing Research Group, Faculty of Electrical and Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City 700000, Vietnam

* Correspondence: leanhvu@tdtu.edu.vn

Abstract: Recent advancements in autonomous mobile robots have led to significant progress in area coverage tasks. However, challenges persist in optimizing the efficiency and computational complexity of complete coverage path planner (CCPP) algorithms for multi-robot systems, particularly in scenarios requiring revisiting or a double pass in specific locations, such as cleaning robots addressing spilled consumables. This paper presents an innovative approach to tackling the double-pass complete coverage problem using an autonomous inter-reconfigurable robot path planner. Our solution leverages a modified Glasius bio-inspired neural network (GBNN) to facilitate double-pass coverage through inter-reconfiguration between two robots. We compare our proposed algorithm with traditional multi-robot path planning in a centralized system, demonstrating a reduction in algorithm iterations and computation time. Our experimental results underscore the efficacy of the proposed solution in enhancing the efficiency of area coverage tasks. Furthermore, we discuss the implementation details and limitations of our study, providing insights for future research directions in autonomous robotics.

Keywords: inter-reconfigurable robot; complete coverage path planner; Glasius bio-inspired neural network; double-pass coverage; multi-robot path planning

MSC: 68T40; 68T07; 65K10

Citation: Sang, A.W.Y.; Yang, Z.; Yi, L.; Moo, C.G.; Mohan, R.E.; Le, A.V. Inter-Reconfigurable Robot Path Planner for Double-Pass Complete Coverage Problem. *Mathematics* **2024**, *12*, 902. https://doi.org/10.3390/math12060902

Academic Editor: Daniel-Ioan Curiac

Received: 17 February 2024
Revised: 14 March 2024
Accepted: 15 March 2024
Published: 19 March 2024

1. Introduction

Robot path planners enable robots to negotiate spaces and adapt to environmental changes based on their objectives. When multiple robots work in the same environment, decision making is needed for robots to work together. Some examples of path planners are seen in individual robots [1,2], robot teams [3,4], reconfigurable robots [5,6], and swarms [7,8].

One class of robots, reconfigurable robots, is able to perform reconfiguration to tackle challenges that autonomous mobile robots are unable to. Reconfiguration can be classified into intra-, inter-, and nested reconfiguration [9]. Intra-reconfigurable robots are able to change their morphologies to tackle challenges such as accessing tight spaces in complete coverage path planning (CCPP). However, these intra-reconfigurable robots are constrained to their joint limits [10] and configuration states [11]. Decision making has to be embedded into the intra-reconfigurable robot path planners [12,13] to account for the robot's joint limits and configuration states. Inter-reconfigurable robots are able to extend their physical limits by combining with another inter-reconfigurable robot. Combining multiple units of an inter-reconfigurable robot [5] extends the robot's physical capabilities, such as an increased payload. However, little or no work has been carried out for path planning for inter-reconfigurable robots.

Path planners were developed to tackle several problems, such as the traveling salesman problem. Early developers of inter-reconfigurable robots have proposed several autonomous functions using the optimized ant colony algorithm [14], reinforcement leaning [15], distributed hedonic coalition formation [16] and many other sampling-based methods [17–20]. These works have similar computation complexities compared to robot swarms [21] and multi-robot systems [22]. Reconfigurable robots with centralized planners have suffered from inter-reconfiguration.

For area coverage in large spaces, multiple autonomous mobile robots are used to perform area coverage successfully. As the environment gets extremely large, the computational requirements to generate CCPP for an increasing number of autonomous mobile robots performing in polynomial time change [23]. Despite the increasing number of autonomous mobile robots, the computation time can be reduced by inter-reconfiguration. In addition, area coverage efficiency is not compromised as the combined robots are able to cover more space, as seen in Figure 1a,b.

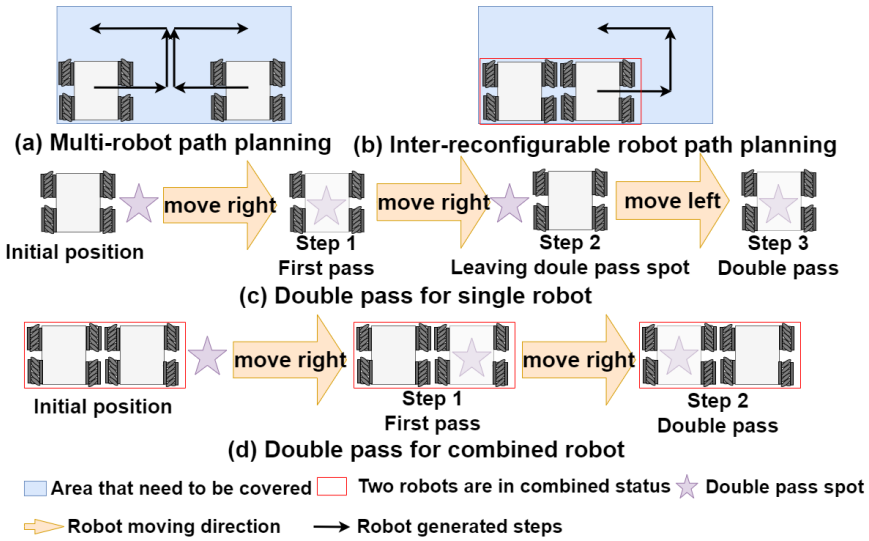

Figure 1. (**a**) A dual-robot system with its zig-zag algorithm paths. (**b**) The path can converge into a single path line. (**c**) How a single robot covers a double-pass area. (**d**) How a combined inter-reconfigurable robot covers a double-pass area.

For the generalized CCPP problem, a bio-inspired neural network (BNN) algorithm was proposed to handle the CCPP problem of multiple robots [24]. This algorithm allows multiple robots to avoid obstacles, escape deadlocks, and achieve complete coverage. However, for the BNN, the work efficiency is quite low, as the amount of calculation is extremely large. The Glasius bio-inspired neural network (GBNN) algorithm is an improved BNN algorithm [25]. It is a rule-based neural network that does not require pretraining or training for its function.

In the context of a real-world problem, a rising key point in CCPP [26–28] requires double-pass complete coverage (DPCC). For example, in a restaurant, most parts of a typical dining floor only require standard cleaning, while some areas might require more attention [29]. In another example, during an inspection task, faulty facility features may be misclassified [30,31], and repeated coverage behaviors can reduce the chance of false positives. To tackle the key points mentioned above, CCPP requires a double pass in the areas that require additional attention after the first coverage. As seen in Figure 1c,d, a single robot takes three steps to double pass an area, while an inter-reconfigurable robot only takes two steps. Using an inter-reconfigurable robot is a more efficient way to tackle the DPCC problem, but this is not addressed in the current state of the art.

This paper proposes a GBNN-based inter-reconfigurable robot path planner for DPCC between two robots. The proposed path planner enables decision making when two

inter-reconfigurable robots combine or split during the DPCC task. The summarized contributions are as follows:

- A proposed path planner for dual inter-reconfigurable robots tackling the DPCC problem using a modified GBNN algorithm;
- The proposed algorithm reduces the computation time compared with state-of-the-art CCPP;
- Demonstration of the proposed algorithm with two inter-reconfigurable robots.

The structure of this paper is as follows. Section 2 describes the proposed path planner for DPCC for inter-reconfigurable robots. Sections 3 and 4 describe the simulations and implementation performed to evaluate the performance of the proposed algorithm compared with a state-of-the-art multi-robot CCPP algorithm. Section 5 concludes with the contributions and describes future works.

2. Double-Pass Coverage for Inter-Reconfigurable Robots

In the GBNN algorithm, the next waypoint in CCPP trajectory is determined by the highest neural activity, and each robot assumes one grid in the bio-inspired neural (BNN) network. However, in the proposed algorithm, the robot in the combined state assumes that there are two grids in the neural network, resulting in modifications to the calculation of the next waypoint position, which is based on two possible positions. The entirety of Section 2 serves as a detailed introduction to the modified algorithm, supplemented with explanations of the rationale behind the changes and their implications for achieving double-pass coverage with inter-reconfigurable robots. An overview of the proposed algorithm is presented in a flowchart, shown in Figure 2. The proposed solution includes an additional level of neural activity to represent areas with double-pass requirements, described in Section 2.1. The proposed path planner decides the inter-reconfiguration state of the robot based on the condition, as described in Section 2.2, and generates neighbors and the next waypoint, as described in Sections 2.3 and 2.4, respectively.

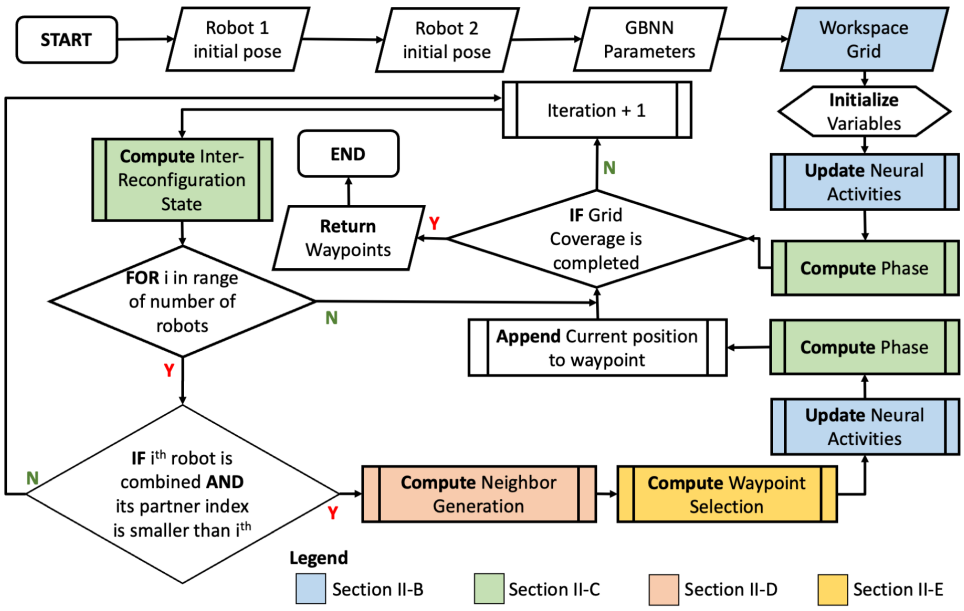

Figure 2. An overview of the algorithm flowchart. The flowchart presents the main flow of the algorithm, key decision-making junctions, and essential functions for the proposed solution.

2.1. Modified Neural Activity Graph

for the above-mentioned changes, Figure 3 illustrates the difference between the typical representation of a GBNN [25] and the proposed modified GBNN. For the modified GBNN,

the neural activity is −1, 1, and 2 when the vertex is marked as obstructed, uncovered, or double pass, respectively.

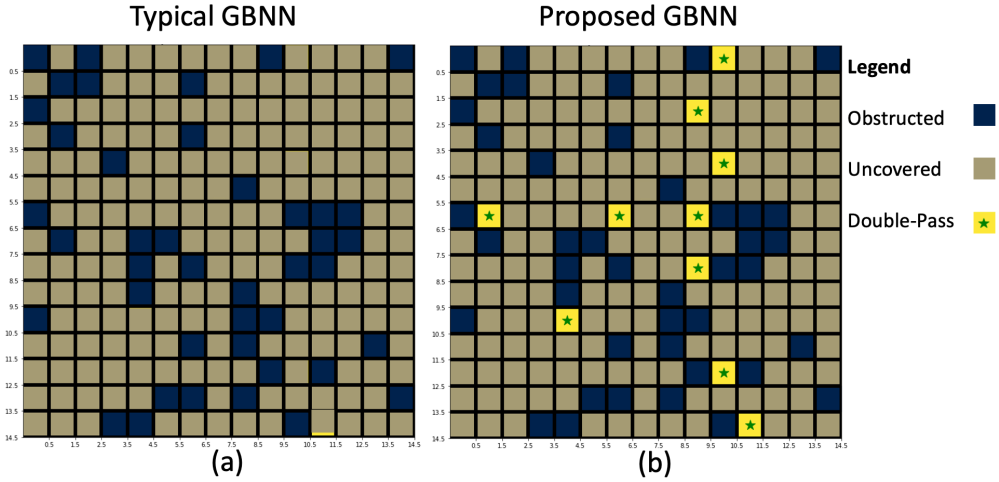

Figure 3. (**a**) GBNN [25]. (**b**) Modified neural activity graph.

Each vertex in the graph represents a neuron in the GBNN [25]. The change in neuron activity with respect to time in each vertex is described as follows:

$$n_i(t+1) = \begin{cases} f(\sum_{j=1}^{m} w_{ij} n_j^+ + c_i), n_i \neq 2 \\ 2 \end{cases} \quad (1)$$

where n_j^+ is the j^{th} neighboring neuron with a non-negative neural activity value, as seen in Figure 4, such that $n_j^+ = max(n_j^+, 0)$. When the neural activity equals two, the neural activity maintains consistency at two and continues to represent the double pass. Meanwhile, w_{ij} represents the coefficient relationship between the i^{th} and its neighboring neurons as follows:

$$w_{ij} = \begin{cases} \exp(-q(i-j)^2), 0 < i-j \leq R \\ 0 \qquad\qquad\quad , i-j > R \end{cases} . \quad (2)$$

The variable R represents the receptive field of all neurons on the grid, as seen in Figure 4. The variable q is a positive constant. The value of q is not a fixed value and can be tuned based on the experimental trials. Then, we set q to be a positive constant. It is noted that the value of q is 2.0 in the simulation. The equation presents how neighboring neural activities affect a node's neural activity level as the first term of (1). On the other hand, (3) presents the value of the second term c_i.

$$c_i = \begin{cases} +E \ , i = \text{Uncovered} \\ -E \ , i = \text{Obstacle} \\ 0 \ \ , i = \text{Covered} \end{cases} \quad (3)$$

where the variable E is recommended to be a large positive constant. The value of E is also not a fixed value and can be tuned. In the simulation, we set the value of E to be 100 based on the experimental trials. In (1), the function will create a local propagation of neural activities' influence. On a global scale, the local propagation will create a gradient of neural activity across the grid:

$$f(x) = \begin{cases} -1 & , x < 0 \\ ax & , 0 \leq x < 1 \text{, where } a > 0 \\ 1 & , x > 1 \end{cases} \quad (4)$$

where a is a small positive constant and (4) limits the neural activities of the non-double-pass area in the range from -1 to $+1$. We set the value of a to 0.01 in the simulation based on the experimental trials.

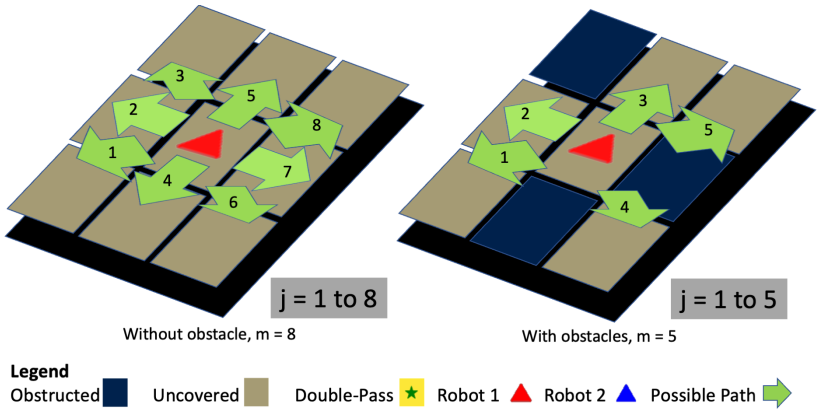

Figure 4. The ith neuron surrounded by the jth neuron in the receptive field $R = 2$.

In each algorithmic iteration, the algorithm updates the neural activities of all neurons. The proposed algorithm uses the properties of the GBNN to generate paths for two inter-reconfigurable robots. Before the generation of the robot paths, the inter-reconfiguration state needs to be determined through the inter-reconfiguration conditions, which are described next.

2.2. Inter-Reconfiguration Conditions

At the start of each iteration, the algorithm will first have to decide whether the robots are independent in a multi-robot or fused state. A proposed example of the inter-reconfiguration decision between two inter-reconfigurable robots, R_1 and R_2, is shown in Algorithm 1. The input and output, R_1 and R_2, respectively, are described by a simple array $(R_i \ni Pose, Config)$, where $Pose$ contains the pose information and $Config$ represents the index of the paired robot in the fused state or is zero when in the multi-robot state.

Algorithm 1 Inter-reconfiguration decision making.

Input: $(R_1 \ni Pose, Config)$, $(R_2 \ni Pose, Config)$, $Path$
Output: R_1, R_2

1: $Trace = set(Path_{n-10}, Path_{n-9}, ..., Path_n)$
2: **for** i in $range\ of\ 2$ **do**
3: $C_1 \leftarrow$ **If** R_1 *is* 1 *grid length from* R_2
4: $C_2 \leftarrow$ **If** *current Phase is Phase* 1
5: $C_3 \leftarrow$ **If** $len(Trace) < 3$
6: **if** $(C_1\ AND\ C_2\ AND\ C_3) == TRUE$ **then**
7: $R_1[Config],\ R_2[Config] = 2,\ 1$
8: **end if**
9: **end for**
10: **return** R_1, R_2

Algorithm 1 is a rule-based algorithm that combines robots mainly based on proximity, the phase of the DPCC problem, and a deadlock check, as seen in C_1, C_2, and C_3 in lines 3–5 of Algorithm 1, respectively. The proximity criterion C_1 combines the robots only when

close and does not require the robots to navigate to each other to combine, ensuring that the computational utilization is minimal.

The operation phase, as seen in Figure 2, is visualized by the graph presented in Figure 5, where two sets of color maps on each node represent the proposed algorithm's phase 1 and 2 for tackling the DPCC problem. Phase 1 is defined when a double-pass coverage area is present and is otherwise set to transit to phase 2 based on other criteria. The purpose of phase 1 is to enable inter-reconfiguration into the fused state during the early stage of DPCC, where a fused inter-reconfigurable robot is assumed to have higher coverage efficiency. In phase 2, the remnants of the DPCC uncovered nodes are usually dispersed, and the inter-reconfigurable robot is split to divide and cover the area in the multi-robot state.

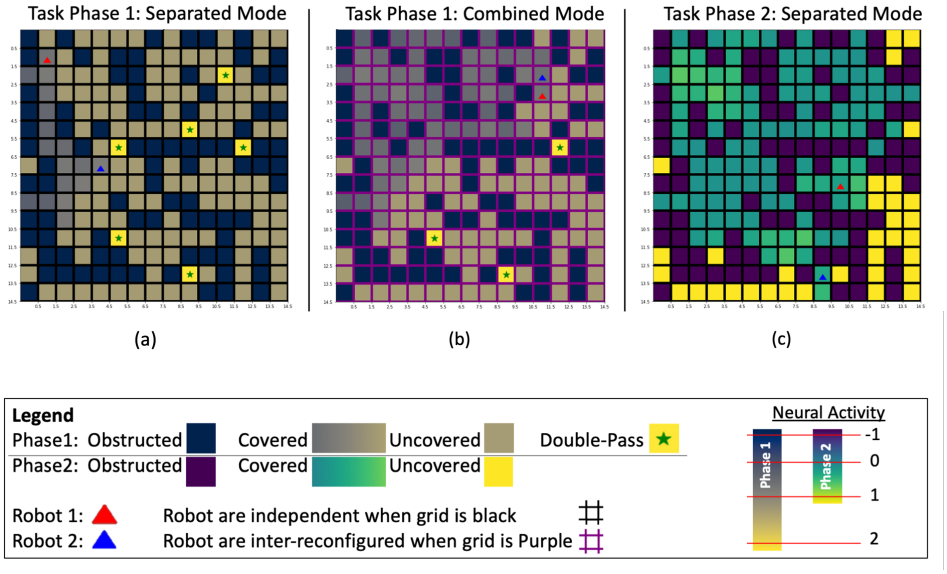

Figure 5. (**a**) GBNN [25]. (**b**) Proposed GBNN. (**c**) Phase 2. In the proposed algorithm, neural activity is -1, 1, or 2 when the vertex is marked as obstructed, uncovered, or double pass, respectively.

In the deadlock check, if there are only two unique waypoints in the past 10 iterations, then it is considered a deadlock, as seen in lines 1 and 5 of Algorithm 1. After deciding the inter-reconfiguration state, generation of the current position neighbors within the receptive field is required to select the next waypoint.

2.3. Neighbor Generation for Inter-Reconfigurable Robots

In the multi-robot state, neighbor generation follows Figure 4, where up to eight neighbors are generated. The fused state of the two inter-reconfigurable robots consists of the R_1 and R_2 positions. The fused state neighbor generator will search for two non-obstacle occupancies as a possible neighbor pair. To generate the list of neighbor pairs for the fused state, the current position of R_1, represented by the red triangle in Figure 6, will search for the position of its neighbors pt_1 in its receptive field where $R = 2$, similar to Figure 4. The current position of R_2, pt_2, is transformed based on pt_1, as seen in Figure 6, based on the transformation rules described by Algorithm 2, where T_x and T_y represent the x and y translation transforms, respectively.

The transform model in Algorithm 2 is a chain of conditions. If the selected neighbor neuron of the current position of R_1 is diagonal, then the condition from line 4, "$dtx \times dty$ *is not* 0, is $TRUE$". It computes the R_2 position transform from R_1 with the logic given in lines 6–10 of Algorithm 2. Otherwise, it will move with a pure translation action ($Tx, Ty = dtx, dty$). As seen in Figure 6, the fused state of R_1 and R_2 can generate up to eight possible neighbor pairs as the possible next waypoint for R_1 and R_2. The next waypoint is selected from the generated neighbor position.

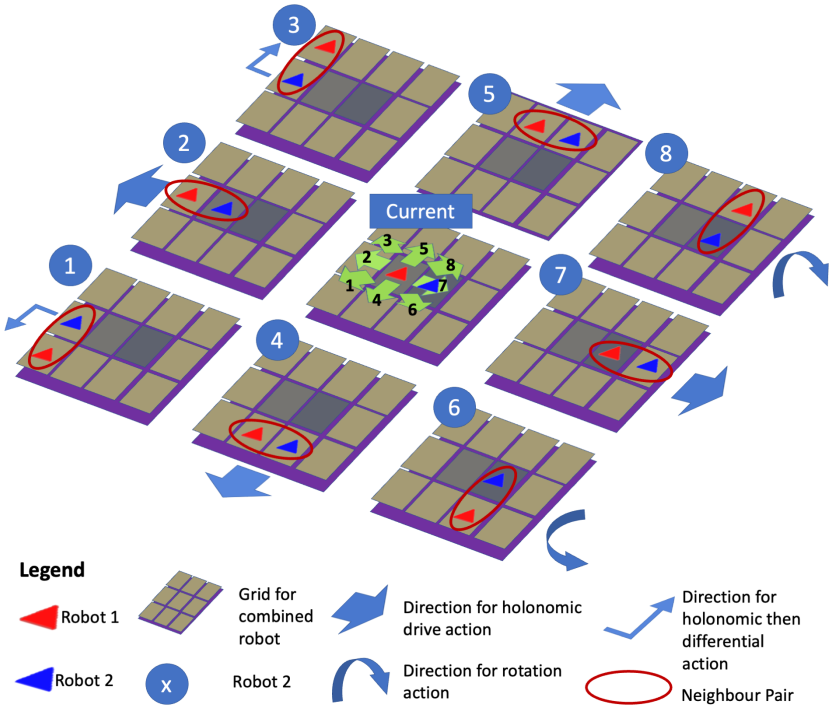

Figure 6. Eight possible neighbor pairs for an inter-reconfigurable robot in the fused state, containing the possible positions for R_1 and R_2.

Algorithm 2 R_2 Transform from R_1.

Input: pt_1, pt_2, *neighbors*, *grid*
Output: *transform*

1: $dtx \leftarrow pt_{1x} - pt_{2x}$
2: $dty \leftarrow pt_{1y} - pt_{2y}$
3: $T_x, T_y \leftarrow 0, 0$
4: **if** $dtx \times dty$ *not* 0 **then**
5: $T_x, T_y \leftarrow 0, 0$
6: **if** dtx *is* 2 *grids length away* **then**
7: $T_x \leftarrow int((dtx)/|dtx) \times 2|$
8: **else if** dty *is* 2 *grids lengthaway* **then**
9: $T_y \leftarrow int((dty)/|dty) \times 2|$
10: **end if**
11: **else if** $dtx + dty$ *is not* 0 **then**
12: $Tx, Ty \leftarrow dtx, dty$
13: **end if**
14: **return** Tx, Ty

2.4. Next Waypoint Selection for Inter-Reconfigurable Robots

Upon generation of neighbors, the next waypoint selection is generated depending on the inter-reconfigurable robot state. Based on the inter-reconfiguration state, the next waypoint, wp_{i+1}, for the inter-reconfigurable robot is as follows:

$$wp_{i+1} = \begin{cases} max(n_k), & \text{multi-robot state} \\ max(n_{k,1} + n_{k,2}) & \text{, fused state} \end{cases} \quad (5)$$

where k represents the elements in the neighbors, generated as represented by the yellow arrows in Section 2.3. In the multi-robot state, the next position is k with the highest neural activity seen in lines 8 and 11 of Algorithm 3. In the fused state, the next neighbor pair is the k neighbor pair with the highest neural activity sum for R_1 and R_2, as seen in lines 4–6

and 11 of Algorithm 3. Here, wp is an array, and in the multi-robot state, it outputs a single position for the robot R_i and, in the fused state, two positions for both R_1 and R_2.

Algorithm 3 Waypoint selection.

Input: *neighbors*, *grid*
Output: wp

```
Initialise cost = [ ];
for k in neighbors do
    x1, y1 ← int(j[0]_x), int(k[0]_y)
    if Robots are Combined then
        x2, y2 ← int(k[1]_x), int(k[1]_y)
        cost.append(grid[x1, y1] + grid[x2, y2])
    else
        cost.append(grid[x1, y1])
    end if
end for
wp ← neighbors(MAX.GetIndex(N_cost))
return wp
```

The proposed algorithm continues iteration until the complete area coverage path is output. Next, the simulation is described.

3. Simulation

The proposed algorithm was compared against the original GBNN [25] for two robots to evaluate the computation time performance. This section briefly describes the simulation set-up, results, and discussion.

3.1. Simulation Set-Up

The simulation was performed on a MacBook Pro M1 with 8 CPU cores. The integrated development environment used to run the simulation was Spyder 4.1.5. As there was no fair comparison between the proposed algorithm and the state-of-the-art solutions, a GBNN would be used as a part of the comparative study. In the proposed algorithm and GBNN [25], the initialized parameters were 100, 1.4, 2.0, 0.7, and 0.01 for E, r, q, and a, respectively. In all simulations, the two inter-reconfigurable robots, R_1 and R_2, were initiated in the top left, as seen in Figure 7.

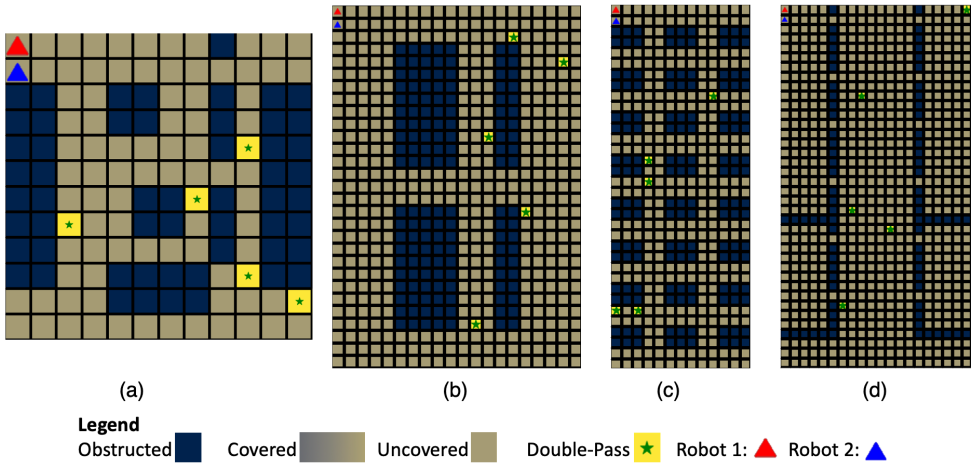

Figure 7. Simulation test environments. The robots are shown in their respective initial positions for all simulations: (**a**) scenario 1, (**b**) scenario 2, (**c**) scenario 3, and (**d**) scenario 4.

Each generated simulation would be evaluated based on the computation iterations, the percentage of iterations generated in the fused state in the overall path plan, and the

time taken per computation iteration in the fused and split configurations (refer to Table 1). Simulations 1 and 2 would be performed to evaluate the proposed algorithm, and they are described next.

Table 1. Operating definitions of terms used in Section 3.

Construct	Operating Definition
Path Length	Total number of waypoints for one robot
Iteration	Generation of a waypoint in the fused or multi-robot state
Iteration Win Rate	Percentage of simulations conducted where computation time of the proposed algorithm is less than that of the GBNN
Fused Ratio	Percentage of simulation, in terms of path length, where the robots are combined
Fused Iteration	Generation of a waypoint in the fused state
Split Iteration	Generation of a waypoint in the multi-robot state for the proposed algorithm or GBNN

3.2. Simulation Results

3.2.1. Simulation 1

Five double-pass areas were randomly generated in four environments, as seen in Figure 7. In each environment, the simulation was iterated 1000 times, and its average win rate of the proposed algorithm against the state-of-the-art GBNN, in terms of iterations and time per fused iteration and split iteration, is shown in Table 2. Env a, b, c, and d in Table 2 represent the environments in Figure 7a–d, respectively. Refer to Table 1 for the operating definition of win rate.

Table 2. Simulation 1 results.

Env	Iteration Win Rate (%)	Fused Ratio (%)	Time, Fused Iteration (ms)	Time, Split Iteration (ms)
a	74.2	33.890	0.0785	0.0568
b	62.5	33.796	0.0912	0.0659
c	66.3	32.911	0.0826	0.0609
d	62.3	28.443	0.0928	0.0667

3.2.2. Simulation 2

We varied the percentage of the double-pass area in Figure 7a to analyze the relationship between the number of iterations from the state-of-the-art GBNN and the proposed algorithm. The fused ratio is shown and summarized in Table 3.

Table 3. Simulation 2 results.

Double Pass	Proposed Algorithm Number of Iterations	State of the Art Number of Iterations	Fused Ratio (%)
0%	140	198	35.7
10%	183	206	21.9
20%	193	160	28.5
30%	150	152	33.3

3.3. Simulation Discussion

Simulation 1's results show that the proposed algorithm for inter-reconfigurable robots required fewer iterations than the multi-robots in the GBNN [25] algorithm in a double-pass

problem. The performance improved from 62.3% to 74.2% of the algorithm's running time in the 1000 simulations performed in all tested environments. The iterations are provided to gauge time, as the time performances varied across different machines. Although the fused iteration took a slightly longer duration to compute than the split iteration, the outcome of the fused iteration was double that of the split iteration. Each fused iteration produced one waypoint for each robot (i.e., R_1 and R_2), while one split iteration generated one waypoint for one robot (i.e., R_i). From the results, the average time was approximately 0.0785–0.0928 ms for a fused iteration. On the other hand, it took 0.0568–0.0659 ms per split iteration, which was equivalent to 0.1136–0.1318 ms for two waypoints for both R_1 and R_2. This shows that the proposed algorithm in the fused state was able to improve the computation time to generate waypoints for R_1 and R_2 18.3–40.4% faster than the state-of-the-art multi-robot GBNN. It is noted that in the proposed algorithm multi-robot state, the algorithm was the same as the multi-robot GBNN and had a similar computation time.

In simulation 2, as seen in Table 3, there were no clear effects for how the number of double passes affected the performance of the coverage algorithms. However, as seen in simulation 1, the environment heavily affected the algorithm. This section has shown the proposed algorithm's ability to tackle the DPCC problem using two inter-reconfigurable robots and demonstrated a reduction in computation time compared with the multi-robot GBNN.

In both simulations, an additional condition was set to allow the robots to transit to phase 2 if 60% area coverage was achieved. This is an empirically obtained number that works best against the multi-robot GBNN, but the optimization of the transition point between phases will not be part of the scope of this study.

4. Implementation

The proposed algorithm is applicable to any type of task that requires double-pass complete coverage (DPCC), such as floor cleaning or inspection activities. Implementation of the proposed algorithm to tackle the DPCC problem was conducted with an inter-reconfigurable robot: Wasp [5].

4.1. Inter-Reconfigurable Robot Wasp

Wasp [5] is an inter-reconfigurable robot that can combine with another Wasp unit to form a bigger form factor and cover a larger area than one standalone unit. The latching module is on the left and right side of the robot chassis, and both Wasp units need to face the same direction for robot-robot docking, as seen in Figure 8. Figure 8a shows two Wasp units that are not oriented in the same direction to combine. The precondition of combining two Wasp units is seen in Figure 8b, where both units are in the same orientation to combine into the fused state as seen in Figure 8c.

(a) Step 1.1. (b) Step 1.2 (c) Step 1.3

Figure 8. Docking process of two Wasp units into a fused state.

4.2. Implementation of the Proposed Algorithm with Wasp

Figure 9 highlights the implementation of the proposed algorithm on two Wasp unites, which shows the experimental image and its corresponding inflated cost map. In the workspace set-up, the robot 1 cross on the floor seen in Figure 9 is an artificially inflated obstacle [32]. The two units of the inter-reconfigurable robot Wasp started in the top-left

corner of the workspace in a different orientation, as seen in iteration 0, and were initialized in the multi-robot state. The robots moved holonomically from iteration 0 to iteration 1. In iteration 1, the proposed algorithm met the inter-reconfiguration conditions to combine, where the robots had to perform automatic docking. Wasp performs automatic docking as seen in steps 1.1, 1.2, and 1.3 in Figure 8a–c, respectively. Then, the robot moved to the next waypoint upon completing physical docking as an inter-reconfigured fused robot, as seen in iterations 4 and 9. A double-pass zone was cleared with a horizontal move and then vertical sequentially. In iteration 10, the fused robot rotated based on the path plan. Finally, the robot moved through the narrow passage in iteration 16. At iteration 16, it achieved 60% complete coverage, and the phase moved from phase 1 to phase 2, triggering the inter-reconfiguration conditions and entering phase 2. The robots then cleared the remaining uncovered spaces independently. The proposed algorithm took 31 iterations, while the GBNN took 34 iterations.

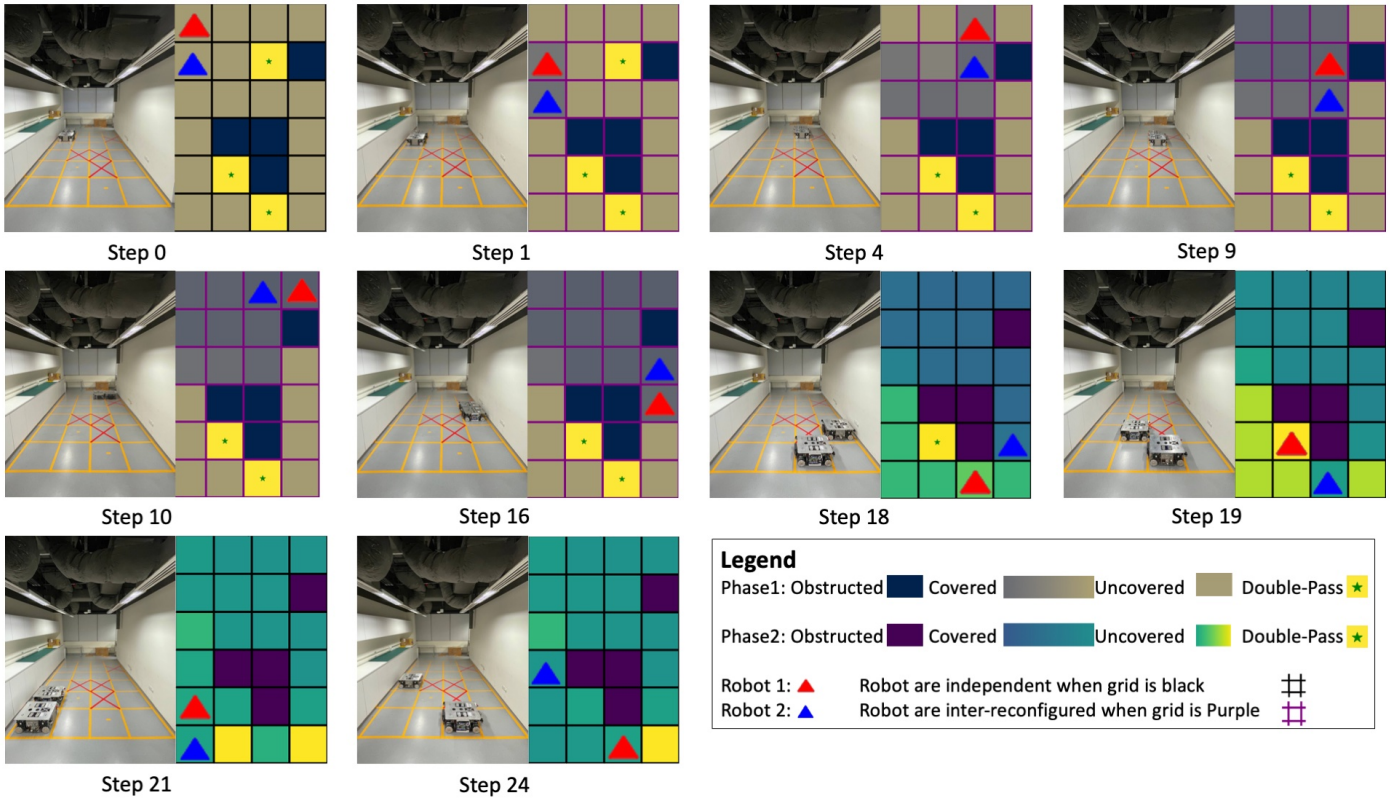

Figure 9. Simulation test environment. The robots are shown in their respective initial positions for all simulations.

4.3. Current Limitations of Proposed Algorithm

The proposed path planner has yet to consider the orientation of the robots. As seen in Section 4.1, not all inter-reconfigurable robots are able to perform docking in all directions. The proposed algorithm also requires an inflated cost map in implementation. When moving toward iteration 16 in the implementation, as seen in Figure 9, the robot partially crossed the artificially inflated obstacle area, where the lethal cost [33] was assumed to be low. More importantly, the path planner does not violate the expectations of global path planning and only selects obstacle-free areas as part of the path plan. The limitations mentioned in this section introduce considerations for implementation. Local planners and automatic docking systems should be considered to enable the utilization of the proposed algorithm. For example, proper tuning of inflation is required during implementation of the proposed algorithm. In this section, the implementation of the algorithm with two inter-reconfigurable Wasp robot was demonstrated.

5. Conclusions and Future Work

This work introduces a novel modified Glasius bio-inspired neural network (GBNN) algorithm tailored to dual inter-reconfigurable robots to address the DPCC problem. The proposed algorithm stands out as the first to incorporate inter-reconfiguration conditions into decision-making processes between two inter-reconfigurable robots, generating complete coverage paths with inter-reconfiguration states. The simulations demonstrated the superior performance of the proposed algorithm, outperforming the GBNN by up to 74.2% in a given environment. Moreover, the proposed algorithm achieved significant computational efficiency improvements, reducing computation by up to 40.4% per iteration compared with traditional multi-robot iterations. Beyond its immediate applications, such as enhanced payload capacity, future work will focus on refining inter-reconfiguration models for various applications and supporting systems, including automatic docking and local planning for inter-reconfigurable robots. Additionally, there is potential to extend this research to explore similar approaches for multiple robots beyond two.

Author Contributions: Conceptualization, A.W.Y.S. and A.V.L.; data curation, A.W.Y.S. and C.G.M.; formal analysis, A.W.Y.S., Z.Y. and L.Y.; methodology, A.W.Y.S., Z.Y. and L.Y.; project administration, R.E.M.; resources, C.G.M. and R.E.M.; software, A.W.Y.S., Z.Y., L.Y. and A.V.L.; visualization, Z.Y. and L.Y.; writing—original draft, A.W.Y.S., Z.Y. and L.Y.; writing—review and editing, C.G.M., R.E.M. and A.V.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Robotics Programme under its National Robotics Programme (NRP) BAU, Ermine III: Deployable Reconfigurable Robots, Award No. M22NBK0054 and also supported by A*STAR under its "RIE2025 IAF-PP Advanced ROS2-native Platform Technologies for Cross sectorial Robotics Adoption (M21K1a0104)" program.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare that there are no conflicts of interest.

References

1. Dugas, D.; Nieto, J.; Siegwart, R.; Chung, J.J. Ian: Multi-behavior navigation planning for robots in real, crowded environments. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 11368–11375.
2. Mohammad, N.; Bezzo, N. A robust and fast occlusion-based frontier method for autonomous navigation in unknown cluttered environments. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; pp. 6324–6331.
3. Li, Q.; Gama, F.; Ribeiro, A.; Prorok, A. Graph neural networks for decentralized multi-robot path planning. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 11785–11792.
4. Kim, M.; Gupta, R.; Sentis, L. CONCERTS: Coverage Competency-Based Target Search for Heterogeneous Robot Teams. *Appl. Sci.* **2022**, *12*, 8649. [CrossRef]
5. Sang, A.W.Y.; Moo, C.G.; Samarakoon, S.B.; Muthugala, M.V.J.; Elara, M.R. Design of a reconfigurable wall disinfection robot. *Sensors* **2021**, *21*, 6096. [CrossRef] [PubMed]
6. Yi, L.; Le, A.V.; Hayat, A.; Elangovan, K.; Leong, K.; Povendhan, A.; Elara, M. Anti-collision static rotation local planner for four independent steering drive self-reconfigurable robot. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 5835–5841.
7. Pfister, K.; Hamann, H. Collective Decision-Making with Bayesian Robots in Dynamic Environments. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; pp. 7245–7250.
8. Alfeo, A.L.; Ferrer, E.C.; Carrillo, Y.L.; Grignard, A.; Pastor, L.A.; Sleeper, D.T.; Cimino, M.G.; Lepri, B.; Vaglini, G.; Larson, K.; et al. Urban Swarms: A new approach for autonomous waste management. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 4233–4240.
9. Tan, N.; Rojas, N.; Elara Mohan, R.; Kee, V.; Sosa, R. Nested reconfigurable robots: Theory, design, and realization. *Int. J. Adv. Robot. Syst.* **2015**, *12*, 110. [CrossRef]
10. Xie, B.; Maciejewski, A.A. Maximizing the probability of task completion for redundant robots experiencing locked joint failures. *IEEE Trans. Robot.* **2021**, *38*, 616–625. [CrossRef]
11. Tan, N.; Mohan, R.E.; Elangovan, K. Scorpio: A biomimetic reconfigurable rolling—Crawling robot. *Int. J. Adv. Robot. Syst.* **2016**, *13*, 1729881416658180. [CrossRef]

12. Do, H.; Le, A.V.; Weeger, O.; Mohan, R.E.; Guo, Y.; Tan, N.N.; Vu, M.B.; Van Duc, P. Path planning for reconfigurable htetro robot combining heat conduction-based and discrete optimization. *IEEE Access* **2021**, *9*, 127019–127036. [CrossRef]
13. Yao, M.; Belke, C.H.; Cui, H.; Paik, J. A reconfiguration strategy for modular robots using origami folding. *Int. J. Robot. Res.* **2019**, *38*, 73–89. [CrossRef]
14. Dai, Y.; Xiang, C.F.; Liu, Z.X.; Li, Z.L.; Qu, W.Y.; Zhang, Q.H. Modular Robotic Design and Reconfiguring Path Planning. *Appl. Sci.* **2022**, *12*, 723. [CrossRef]
15. Song, Q.; Ye, D.; Sun, Z.; Wang, B. Motion planning techniques for self-configuration of homogeneous pivoting cube modular satellites. *Aerosp. Sci. Technol.* **2022**, *120*, 107249. [CrossRef]
16. Dutta, A.; Ufimtsev, V.; Said, T.; Jang, I.; Eggen, R. Distributed Hedonic Coalition Formation for Multi-Robot Task Allocation. In Proceedings of the 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE), Lyon, France, 23–27 August 2021; pp. 639–644.
17. Sun, H.; Li, M.; Song, J.; Wang, Y. Research on self-reconfiguration strategy of modular spherical robot. *Int. J. Adv. Robot. Syst.* **2022**, *19*, 17298806221081665. [CrossRef]
18. Jones, A.B.; Cameron, T.; Eichholz, B.; Loegering, D.; Kray, T.; Straub, J. Self-reconfiguring modular robot learning for lower-cost space applications. In Proceedings of the 2019 IEEE Aerospace Conference, Big Sky, MT, USA, 2–9 March 2019; pp. 1–6.
19. Liu, C.; Yu, S.; Yim, M. Motion Planning for Variable Topology Truss Modular Robot. In Proceedings of the Robotics: Science and Systems, Corvalis, OR, USA, 12–16 July 2020.
20. Whitman, J.; Bhirangi, R.; Travers, M.; Choset, H. Modular robot design synthesis with deep reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 10418–10425.
21. Zhou, X.; Zhu, J.; Zhou, H.; Xu, C.; Gao, F. Ego-swarm: A fully autonomous and decentralized quadrotor swarm system in cluttered environments. In Proceedings of the 2021 IEEE international conference on robotics and automation (ICRA), Xi'an, China, 30 May 2021–5 June 2021; pp. 4101–4107.
22. Das, P.; Jena, P.K. Multi-robot path planning using improved particle swarm optimization algorithm through novel evolutionary operators. *Appl. Soft Comput.* **2020**, *92*, 106312. [CrossRef]
23. Seo, J.; Paik, J.; Yim, M. Modular reconfigurable robotics. *Annu. Rev. Control. Robot. Auton. Syst.* **2019**, *2*, 63–88. [CrossRef]
24. Luo, C.; Yang, S.X. A real-time cooperative sweeping strategy for multiple cleaning robots. In Proceedings of the IEEE International Symposium on Intelligent Control, Vancouver, BC, Canada, 30 October 2002; pp. 660–665.
25. Sun, B.; Zhu, D.; Tian, C.; Luo, C. Complete Coverage Autonomous Underwater Vehicles Path Planning Based on Glasius Bio-Inspired Neural Network Algorithm for Discrete and Centralized Programming. *IEEE Trans. Cogn. Dev. Syst.* **2019**, *11*, 73–84. [CrossRef]
26. Yao, P.; Zhao, Z. Improved Glasius bio-inspired neural network for target search by multi-agents. *Inf. Sci.* **2021**, *568*, 40–53. [CrossRef]
27. Sun, R.; Tang, C.; Zheng, J.; Zhou, Y.; Yu, S. Multi-robot path planning for complete coverage with genetic algorithms. In Proceedings of the Intelligent Robotics and Applications: 12th International Conference, ICIRA 2019, Shenyang, China, 8–11 August 2019; Proceedings, Part V 12; Springer: Berlin/Heidelberg, Germany, 2019; pp. 349–361.
28. Giang, T.T.C.; Binh, H.T.T. Hybrid Boustrophedon and Partition Tree Group Algorithm for Coverage Path Planning Problem with Energy Constraints. In Proceedings of the Recent Challenges in Intelligent Information and Database Systems: 14th Asian Conference, ACIIDS 2022, Ho Chi Minh City, Vietnam, 28–30 November 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 626–640.
29. Pathmakumar, T.; Kalimuthu, M.; Elara, M.R.; Ramalingam, B. An autonomous robot-aided auditing scheme for floor cleaning. *Sensors* **2021**, *21*, 4332. [CrossRef] [PubMed]
30. Rahman, A.; Wu, Z.Y.; Kalfarisi, R. Semantic deep learning integrated with RGB feature-based rule optimization for facility surface corrosion detection and evaluation. *J. Comput. Civ. Eng.* **2021**, *35*, 04021018. [CrossRef]
31. Wilkins, A.H.; Strange, A.; Duan, Y.; Luo, X. Identifying microseismic events in a mining scenario using a convolutional neural network. *Comput. Geosci.* **2020**, *137*, 104418. [CrossRef]
32. [ROS Q&A] 136—How to Edit a Map Generated with Gmapping. Available online: https://www.youtube.com/watch?v=BfCUfmJLJDY&t=160s (accessed on 28 February 2023).
33. Zheng, K. Ros navigation tuning guide. In *Robot Operating System (ROS) The Complete Reference*; Springer: Berlin/Heidelberg, Germany, 2021; Volume 6, pp. 197–226.