Optimizing Multirobot Systems: A Novel Distributed Task Allocation Approach for Maximizing Task Executions

Nayyer Fazal[1*], Muhammad Tahir Khan[1], Javaid Iqbal[2], Shahbaz Khan[3] , Zubair Ahmad Khan[1]

[1] Department of Mechatronics Engineering, University of Engineering & Technology Peshawar, Pakistan
2 National University of Science and Technology ,Islamabad .Pakistan
3 Institute of Manufacturing, University of Engineering and Applied Sciences Swat. Pakistan
Corresponding Author: Nayyer Fazal

Abstract- The present manuscript explores the complex domain of distributed task allocation, with a specific emphasis on optimizing the number of successfully completed tasks in multirobot systems. The inherent challenges arise due to the temporal limitations imposed by task deadlines and the restricted fuel capacities of robotic vehicles, often resulting in the unattainability of successfully completing all tasks. In this proposal, we present a groundbreaking paradigm that introduces the Effective and Efficient Performance Impact (EEPI) algorithm. This algorithm stands out due to its unique cost function and task release methodology, which contribute to its effectiveness and efficiency. The fundamental principles underlying the proposed cost function are centered on two pivotal tenets. The primary objective of this approach is to reduce the duration of travel required for each vehicle to reach their assigned task locations. This optimization is strategically designed to maximize the available time for task execution, taking into consideration the limited fuel resources at hand. Additionally, this approach guarantees the prompt commencement of each task in close proximity to its designated deadline, thereby facilitating the efficient prioritization of tasks with earlier deadlines over those with later deadlines. In order to address the challenges pertaining to the impacts on removal performance and inclusion performance, a strategy is proposed wherein tasks assigned to a vehicle are collectively released if said vehicle has achieved the highest task removal count during the task removal phase. The implementation of this strategic approach effectively enhances the overall quantity of tasks that are executed successfully. The effectiveness of the proposed Energy-Efficient Power Injection (EEPI) algorithm has been thoroughly validated through a combination of comprehensive simulations and real-world hardware-in-the-loop experiments. The effectiveness of the proposed distributed task allocation algorithm in maximizing task execution and its efficiency in reducing the number of iterations and convergence time have been substantiated through comparative analyses against state-of-the-art algorithms.

Index Terms- Task allocation, Distributed algorithm, hardware in loop experiment, multirobot system.

I. Introduction

Over the past few decades, there has been a substantial rise in the use of multirobot systems in many industries, encompassing both civilian and military domains [1]. The utilization of these technologies covers a wide spectrum, including the monitoring of moving objects [2], investigation of particular regions [3], search and rescue missions [4], and endeavors associated with disaster assistance [5]. Efficiently integrating and coordinating various autonomous robotic vehicles is essential for achieving higher levels of effectiveness and efficiency [6]. The main challenge faced in implementing cooperative multirobot systems for practical purposes relates to the domain of task allocation. The goal of task allocation is to efficiently coordinate vehicles to perform tasks and achieve one or more global objectives [7].

The categorization of multirobot systems can be roughly divided into two primary classifications: centralized systems and distributed systems, which are differentiated by their distinct organisational paradigms [8]. Centralized systems employ a server to efficiently gather and merge diverse types of information, such as situational awareness (SA), from numerous vehicles. The server utilizes centralized algorithms [9], to create a detailed job allocation plan for the entire fleet. The suggested methodology seeks to reduce the computing and communication demands placed on individual vehicles, hence improving their compactness and cost-effectiveness. Nevertheless, achieving a globally uniform understanding of the current situation (situational awareness) in real-world scenarios is difficult due to the existence of unpredictable changes and restricted communication capacities. These parameters directly affect the system's mission range and overall durability.

However, it is important to acknowledge that there has been substantial advancement in the domain of distributed multirobot systems and the corresponding methodologies for allocating tasks. The purpose of these developments is to enhance the overall mission range and durability of such systems [10], [11]. Auction approaches have demonstrated efficacy in resolving single-task allocation difficulties, when each vehicle is assigned a solitary task [12]. These techniques include cars putting separate bids for tasks, and then an auctioneer decides the winning bidder for each work. Within the domain of multi-task allocation, scholars have suggested the implementation of combinatorial auction techniques. Nevertheless, it is important to acknowledge that these approaches are inefficient due to the exponential increase in the number of potential task combinations as the number of tasks grows.

The consensus-based bundle algorithm (CBBA) [13] utilizes a systematic approach to address the challenges related to multi-task allocation. This approach incorporates repeated procedures of constructing bundles and reaching an agreement. However, the assignment of tasks becomes increasingly difficult as the proportion of tasks to vehicles grows. To address this difficulty, a new method called the Performance Impact (PI) algorithm was proposed as a potential approach to efficiently manage time constraints [14]. After the initial iterations, further improvements were made, including the implementation of PI-softmax, in order to increase the efficiency of task execution [36]. However, it is important to mention that as the ratio of tasks to vehicles increases, both the Proportional-Integral (PI) and PI-softmax techniques have significant restrictions.

This research study introduces the Effective and Efficient Performance Impact (EEPI) method, which seeks to enhance task execution by maximizing the number of jobs performed effectively without requiring rescheduling. The current study includes several significant advances, specifically the creation of a novel cost function that tries to optimize task execution, and the implementation of a task release method intended to reduce any negative impact on performance validity. The EEPI algorithm has proven to be effective by showing fewer iterations and communications compared to existing approaches. The subsequent sections offer a thorough explanation of the mathematical representation of the distributed job allocation problem. Furthermore, a new cost function is presented, and the EEPI algorithm and work release mechanism are thoroughly examined. The effectiveness and productivity of the EEPI algorithm have been confirmed by simulation results and hardware-in-the-loop trials. These findings provide useful insights for the actual utilisation of the algorithm in many applications.

II. FORMULATION OF PROBLEM

This study focuses on analysing a group of unmanned vehicles, referred to as V = {1, 2, ..., n}. These vehicles are utilised to carry out a specific set of tasks, denoted as T = {1, 2, ..., m}. It is crucial that each vehicle reaches the assigned destination for a task before starting the work, following operating rules. The vehicles employ a decentralised operational method, where a thorough task assignment strategy is collaboratively developed by exchanging information through a communication network. The primary objective of task allocation is to determine the assignment of work to individual vehicles and the organization of these tasks in a way that maximizes overall global goals. The main goal of this study is to enhance task allocations by maximizing the total number of tasks assigned to vehicles, while also maintaining adherence to time constraints and fuel limitations.

Regarding the job allocation problem being discussed, it is crucial to acknowledge that each vehicle can only perform one task at a time. Moreover, it is crucial to recognize that every duty necessitates the sole utilization of a designated vehicle. Furthermore, it is noteworthy that a solitary vehicle possesses the capacity to consecutively execute numerous duties. The use of symbols is followed by a thorough table, referred to as Table I, that offers a detailed explanation and illustration of each symbol. Tasks are classified into distinct categories based on their specific requirements, and only vehicles equipped with the requisite capabilities can do a given work. Each task, denoted by the variable j, is characterized by a pre-established deadline time (sj). The timely initiation of each task is crucial for its proper completion. The variable τj denotes the time taken by a vehicle to execute job j, starting from the commencement of execution until it is successfully finished. It is hypothesized that different tasks, even those that are the same, may have varied durations, while the length remains the same for the same task performed by multiple vehicles. The fuel capacity of each vehicle, indicated by the active time (fi), shows the projected duration until the vehicle depletes its gasoline. The allocation of tasks to vehicle i is restricted by capacity restrictions, where vehicle i can be assigned a maximum of Li tasks. The vehicles commence their mission within the designated region and must return to base B situated outside the mission area before depleting their fuel. This limitation is especially crucial for certain types of vehicles, such as unmanned aerial vehicles (UAVs).

The variable "ai" represents the task list that is linked to vehicle "i". This task list consists of a sequentially arranged set of tasks from the set T. The element represented as aik = j in the sequence ai signifies the kth task carried out by vehicle i within the sequence ai, namely task j. The estimation of the start time for task execution in the field of artificial intelligence, as well as the following return time to the base for a specific vehicle labelled as "i," can be accomplished using a recursive method.

III. DEVELOPED METHODOLOGY

The Effective and Efficient Performance Impact (EEPI) method is specifically intended to systematically attain optimal performance outcomes. The technique operates iteratively, consisting of two separate phases: task inclusion and consensus and task removal. These phases are carried out in an alternating manner until stability is achieved in both phases. The measurement and evaluation of the impacts on removal performance (RPIs) and inclusion performance (IPIs) are crucial in the iterative procedures being considered. The Introducing a New Task Impact on Vehicle Cost (IPI) statistic measures the specific influence of adding a new task on the total cost of a vehicle. The Removal of Task Impact (RPI) statistic measures the specific impact of eliminating a task on the total cost of a vehicle. During the task inclusion step, tasks are added to the task list of each vehicle by assessing the differences between their Relative Priority Indices (RPIs) and Individual Priority Indices (IPIs). During the consensus and task elimination phase, cars participate in the trade of their individual local assignments. Afterwards, each vehicle updates its own assignment by incorporating the assignments obtained from other cars. Subsequently, tasks with lower Relative Priority Index (RPI) values, as decided by other cars, are methodically removed from the task list of each corresponding vehicle. To reduce the frequency of invalid Resource Processing Indices (RPIs) and Invalid Processing Indices (IPIs), a task release method has been created and put into effect after each task removal phase. This technique involves clearing all tasks from the task list of the vehicle that has successfully completed the greatest number of tasks. The cost function is a basic idea in diverse domains of study and analysis. It functions as a quantitative depiction of the costs linked to a specific system, process, or decision. Quantitatively.

Algorithm1: Including Vehicle I in the Task Phase

1. Initialize: Begin the task inclusion phase on Vehicle i.
2. Loop: Continue the process until the total number of tasks in the task list (|ai|) reaches the vehicle's capacity (Li).
3. Compute IPIs: Calculate the Inclusion Performance Impacts (IPIs) for each task based on Equation (11).
4. Evaluate Impact: Determine the maximum difference (g) between the RPI and IPI of tasks.
5. Check Impact: If the impact (g) is positive (g > 0),

- Identify Task: Find the task (q) that contributes the most to the impact.
- Find Position: Identify the optimal position (p) to insert task q in the task list.
- Insert Task: Add task q to the task list at position p.
- Update Assignment: Record the vehicle i as the assigned vehicle for task q (βiq = i).
- Update RPI: Adjust the Removal Performance Impact (RPI) of task q (w⊖iq = w⊕iq).
- Update Costs: Adjust the start time and costs of tasks in the updated task list ai.
- Check Impact: If the impact (g) is non-positive (g ≤ 0), exit the loop.
- Update RPIs: After the task inclusion phase, update the RPIs of tasks in ai using Equation (13).

A. Cost Function

The temporal duration during which a vehicle remains operational, constrained by its fuel capacity, can be divided into two distinct components: the execution time, which denotes the duration required for task completion, and the travelling time, which represents the duration necessary for the vehicle to traverse to different task locations. The optimization of travel time is of utmost importance in order to maximize the efficiency and productivity of task execution. The cost function under consideration incorporates the expenses associated with travelling for each task, with a particular emphasis on the imperative of minimizing travel time. Furthermore, in light of the temporal duration between the initiation of a task and its designated deadline, commonly referred to as the deadline cost, tasks are strategically ranked and organized in accordance with their relative proximity to said deadlines. The primary objective of the cost function is to allocate time slots to tasks that have earlier deadlines, thereby facilitating the completion of a greater number of tasks within the given timeframe.

B. Task inclusion Phase

During the task inclusion phase, candidate tasks undergo thorough evaluation and assessment to determine their suitability for inclusion in the task list of each vehicle. The tasks are inserted one after another in a sequential manner. Then, an evaluation is carried out to determine their influence on both the Resource Performance Indicators (RPIs) and the influence Performance Indicators (IPIs). The inherent process index (IPI) of each task can be influenced by the exact order in which the tasks are included. The procedural protocol is designed to systematically integrate jobs in a way that efficiently reduces the total Resource Performance Index (RPI).

Algorithm 2: In the process of removing tasks from vehicle i,

1. Identify Candidates: Determine the list of candidate tasks (di) intended for removal.
2. Initialize Counter: Set the count of removed tasks (θii) to zero.
3. Loop: Continue the process as long as there are tasks in the candidate list (|di| > 0).
4. Calculate RPIs: Compute the Removal Performance Impacts (RPIs) for tasks in di using Equation (13).
5. Evaluate Impact: Determine the maximum impact (g⋄) based on the differences between RPI and the current Removal Performance Impacts.
6. Check Impact: If the impact (g⋄) is positive (g⋄ > 0),

- Identify Task: Find the task (q) with the highest impact in di.
- Remove Task: Eliminate task q from the task list (ai) and candidate list (di).
- Update Costs: Adjust the start time and costs of the remaining tasks in ai.
- Update Counter: Increase the count of removed tasks (θii) by 1.

7. Check Impact: If the impact (g⋄) is non-positive (g⋄ ≤ 0), exit the loop.
8. Update Assignments: After the task removal phase, confirm the assigned vehicle for each task in the candidate list (di) as vehicle i (βi j = i for all j in di).

C. Consensus and Task Removal Phase

Conflicts may occur among the allocations of different vehicles during the consensus phase. To resolve conflicts, cars exchange their Registered Participant Identifier (RPI) and winner lists, using the consensus criteria set by the Cross-Border Blockchain Alliance (CBBA). During the subsequent step of task elimination, tasks with lower Relative Priority Indices (RPIs), as assessed by other vehicles, are gradually discarded. The task removal step is implemented to maintain the ongoing coherence of tasks with the consensus decisions that have been reached.

Algorithm 3: Program Outline for the Suggested EEPI

Initialization:
Define the world, vehicles, tasks, and network topology.
Vehicle Initialization:
For each vehicle i in the set of vehicles V:
Initialize task list (ai), RPI (w⊕i), IPI (w⊖i), winner list (βi), timestamp list (δi), and removal list (θi).
Iteration Loop:
While convergence criteria are not met, continue the iteration.
Communication and Update Phase:
For each vehicle k in the set of vehicles V:
For each vehicle i in the set of vehicles V:
If vehicles k and i are connected:
Update RPI (w⊖i), winner list (βi), timestamp list (δi), and removal list (θi) based on the information from vehicle k using CBBA rules.

Task Removal and Inclusion Phases:
For each vehicle i in the set of vehicles V:
Execute the task removal phase.
If i is the vehicle that removed the most tasks (arg maxk⩽n θik) and the count of removed tasks (θii) is greater than 1:
Remove all tasks in ai, reset their RPIs, IPIs, and winners.
Execute the task inclusion phase.
Convergence Check:
If the algorithm has converged, exit the iteration loop.
End of Main Program.

D. Proposed Task Release Procedure

A task release method has been established to resolve the problem of invalid Resource Provider Identifiers (RPIs) and Infrastructure Provider Identifiers (IPIs). After the task removal phase is finished, if a vehicle successfully removes the most tasks, it is important to mention that all tasks linked to that vehicle will be instantly freed. The purpose of implementing this technique is to reduce the occurrence of differences in Relative Performance Indexes (RPIs) and Individual Performance Indexes (IPIs) that may result from the removal of tasks. The main element of the EEPI programme entails the establishment of task-specific lists for each vehicle. Subsequently, a sequence of iterative broadcasts of assignments and updates, which comply with the CBBA consensus rules, takes place. This is then followed by further phases for removing and including tasks. The algorithm continues to iterate until a level of stability is achieved in both phases.

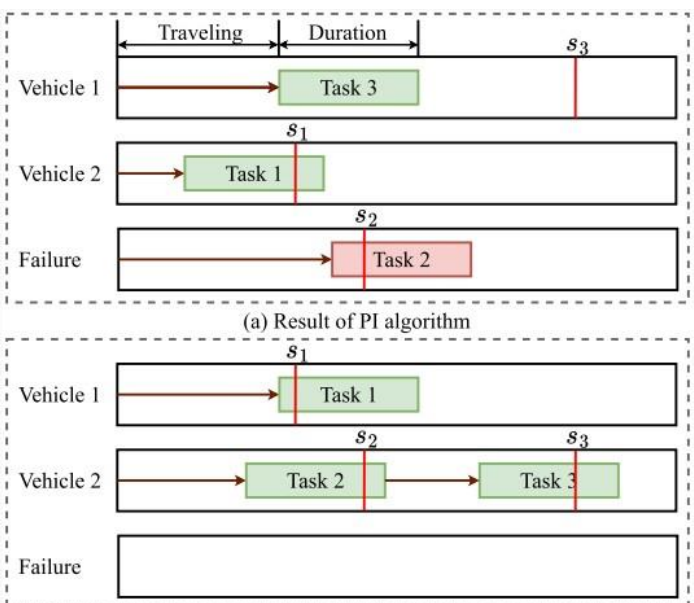

Figure 1. Developed methodology with cost function.

E. Convergence

To address potential issues arising from convergence, especially in algorithms utilising proportional-integral (PI) control, additional constraints are included. The indicated limitations aim to expedite the timely distribution of RPI updates across cars, thus reducing the likelihood of endless cycles and promoting convergence within a limited number of iterations. The EEPI method, as originally suggested, presents a resilient framework that exhibits both efficacy and efficiency in the realm of distributed job allocation.

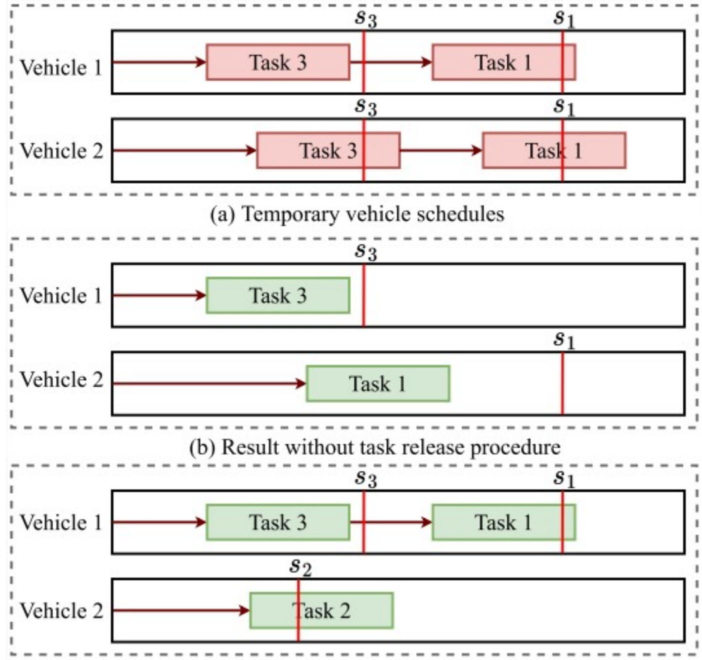

Figure 2. Result with task release procedure.

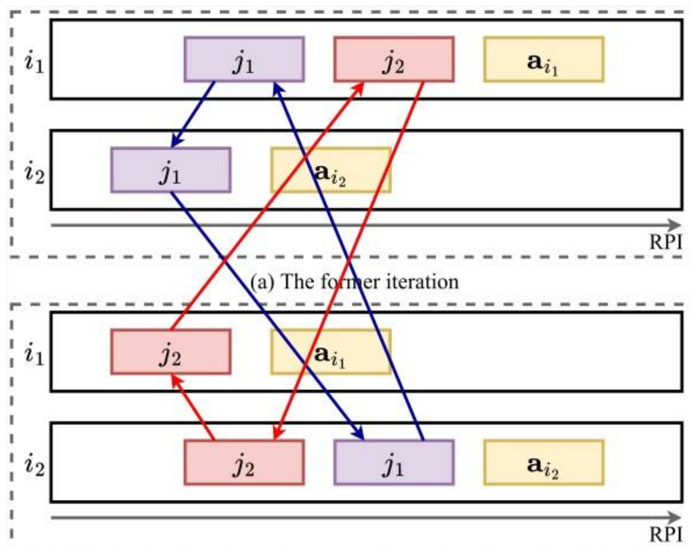

Figure 3. When PI-based algorithms fail, task j1 and task j2's relative RPIs in the idea of cars i1 and i2: The previous iteration. The last iteration.

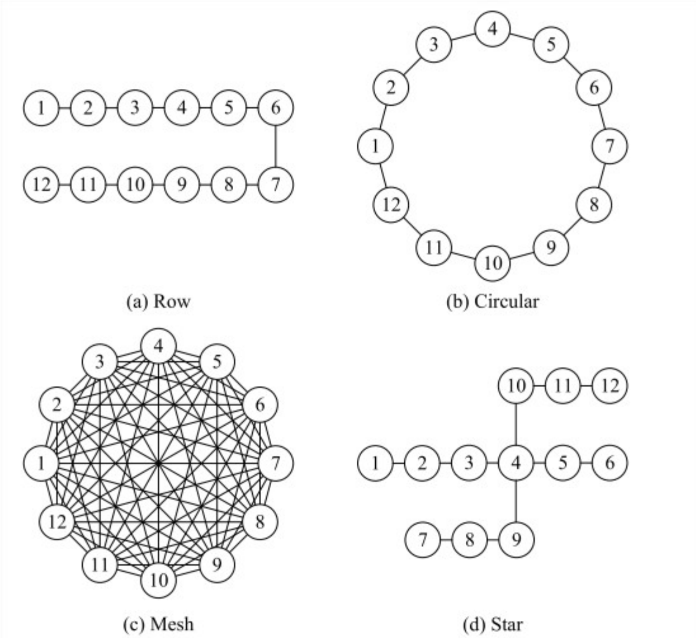

Figure 4. Simulations with 12 vehicles use these communication network configurations. Bidirectional communication is possible between two vehicles connected by a solid line in topologies.

IV. Experimentation

A. Possible Test Cases and Parameter Setups

To verify the suggested Energy Efficient Positioning Indicator (EEPI), a set of simulations were conducted using a search and rescue scenario, as described in reference [15]. In this particular situation, autonomous cars were tasked with the duty of delivering vital aid and support to persons who had experienced a devastating incident. The classification of jobs into two distinct categories, specifically the dispensation of medications and the provision of sustenance, was carried out with the primary aim of matching the individual requirements of survivors. A classification of vehicles was conducted, leading to the recognition of two separate categories: vehicles specifically designed for the conveyance of medical supplies and vehicles specifically designated for the delivery of food products.

Survivors were generated consistently and uniformly within a mission area that measured 10,000 m in length, 10,000 m in breadth, and 1,000 m in height. Simultaneously, automobiles were initially placed on a two-dimensional surface. The vehicles used for transporting medicines and food had cruising speeds of 30 m/s and 50 m/s, respectively. The base station, located near the mission area, served as a designated meeting location for vehicles to return to. The lengths of jobs, estimations of fuel consumption, and deadlines were reliably determined across various parameters.

The communication network topology has been highly important, featuring four various network topologies as shown in the drawings. The study considered the limitations of both communication range and perception range, ensuring that vehicles only engaged in communication activities within their specific operational boundaries. To assess the effectiveness of the proposed Energy Efficiency Performance Index (EEPI), a comparative analysis was performed, involving three established methods: the Consensus-Based Bundle Algorithm (CBBA), the Performance Index (PI), and the Performance Index with Maximum Assignment (PI-maxAss). The research was conducted utilising MATLAB R2017b, a commonly employed software platform for numerical computation and algorithmic development. Each scenario involved doing one hundred separate iterations to evaluate task assignments, the amount of iterations, and CPU time needed to achieve convergence.

B. Results for Simulation

The algorithms underwent evaluation utilising a fleet of 12 and 18 vehicles, which were responsible for the transportation of essential medical supplies and sustenance to a varying number of survivors, ranging from 36 to 132 individuals.

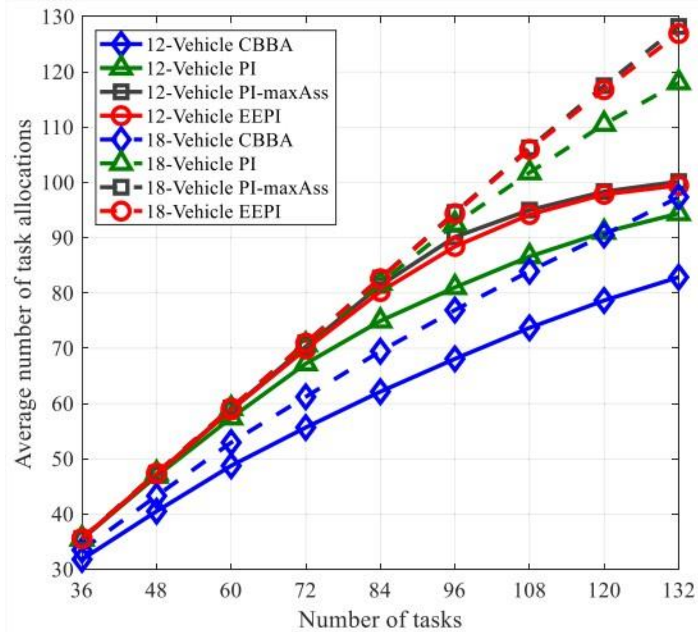

Figure 5. CBBA Average task allocation

The experimental evaluation demonstrates that EEPI consistently exhibited superior performance compared to CBBA and PI. Notably, the achieved results were found to be in close proximity to those obtained by PI-maxAss. The rate of task allocations exhibited a notable acceleration when the number of tasks was relatively low, whereas it demonstrated a deceleration trend as the number of tasks increased, contingent upon the task-to-vehicle ratios.

The experimental results indicate that EEPI outperformed both PI and PI-maxAss in terms of the number of iterations and CPU time. In the context of a high task-to-vehicle ratio, it was observed that EEPI exhibited superior performance in terms of iterations when compared to CBBA. However, it is worth noting that EEPI incurred higher CPU time as a result of its heightened computational complexity.

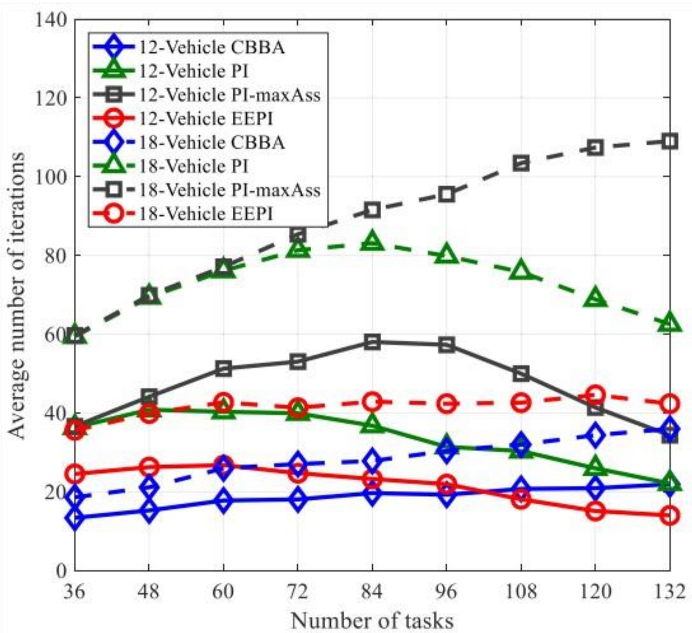

Figure 6. CBBA Average number of iterations

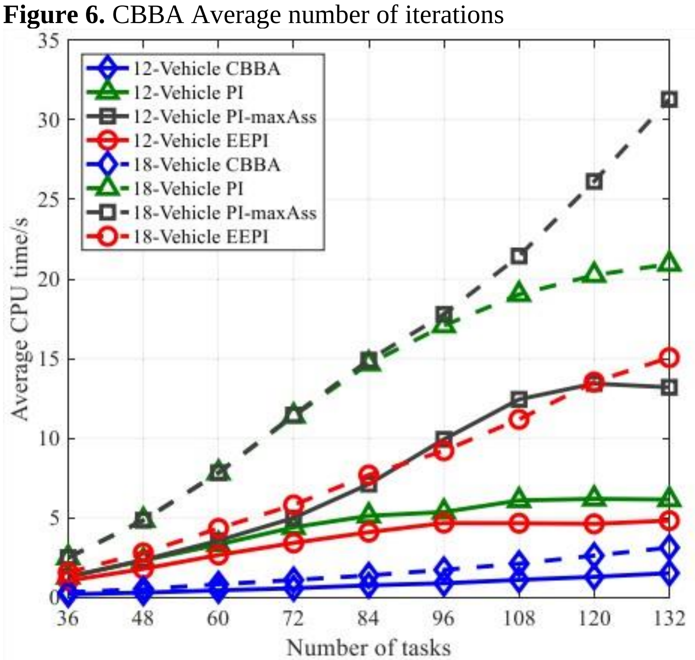

Figure 7. CBBA Average CPU times/second.

C. Network Topologies

The findings exhibited variability across diverse network topologies. The experimental results consistently demonstrated that the EEPI algorithm exhibited superior performance compared to both the CBBA and PI algorithms. Furthermore, the performance of EEPI was found to be comparable to that of PI-maxAss. The convergence speed and CPU time were found to be in favour of the EEPI method when compared to both the PI and PI-maxAss methods.

D. With or Without Releasing Tasks:

The task allocations of the EEPI (Enterprise Efficiency and Productivity Index) demonstrated notable enhancements subsequent to the completion of the consensus and removal phases. The observed trend indicates a marginal increase in the number of iterations, while still maintaining a higher level of efficiency compared to CBBA, PI, and PI-maxAss algorithms.

Table 1. Various network topologies simulation result

Configuration	Topology	Average task allocations				Average iterations			
		CBBA	PI	PI-maxAss	EEPI	CBBA	PI	PI-maxAss	EEPI
Two types	Row	62.56	75.18	81.23	80.39	18.70	35.34	55.75	23.55
	Circular	62.56	75.18	81.23	80.39	18.70	35.34	55.75	23.55
	Mesh	62.55	75.12	81.18	80.05	7.48	16.90	25.78	9.51
	Star	62.57	75.19	81.18	80.35	21.07	41.87	69.29	27.57
Single type	Row	64.05	77.54	82.21	81.62	46.45	97.95	155.47	57.68
	Circular	64.08	77.34	82.16	81.64	33.71	89.21	146.71	42.60
	Mesh	64.07	77.23	82.03	81.37	10.23	22.00	31.27	12.66
	Star	64.08	77.36	82.17	81.66	31.36	72.17	105.83	42.58

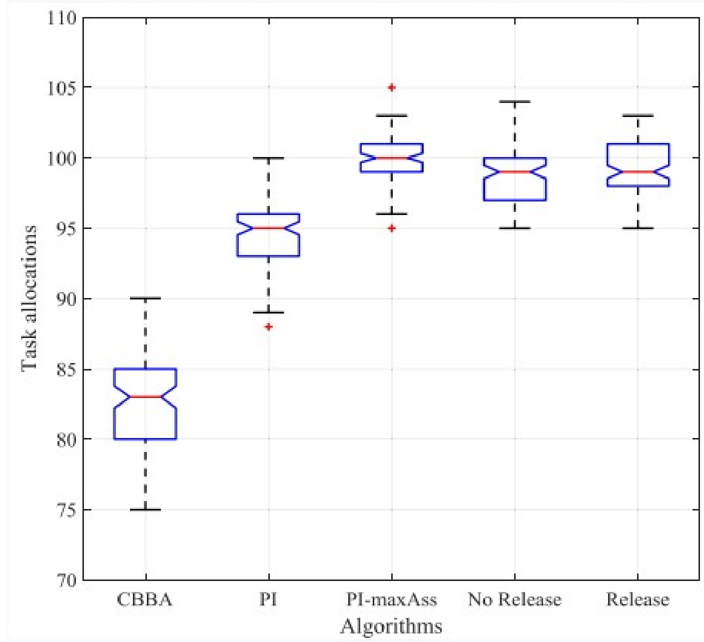

Figure 8. CBBA task allocation box plot.

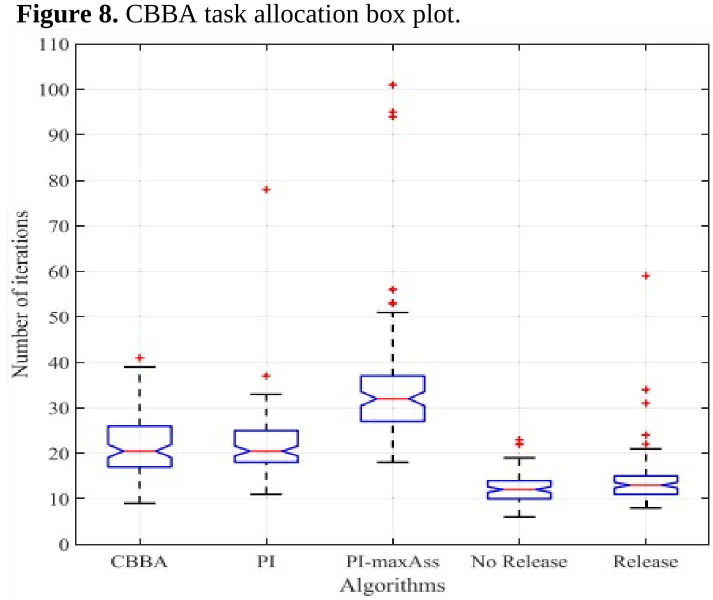

Figure 9. with or without releasing task, number of iteration.

In the realm of task allocations, it has been observed that EEPI consistently exhibits superior performance when compared to CBBA and PI. The observed trend of enhanced task allocations across all algorithms can be attributed to the expanding communication range. The significance of communication in the context of EEPI became apparent when its scope was limited.

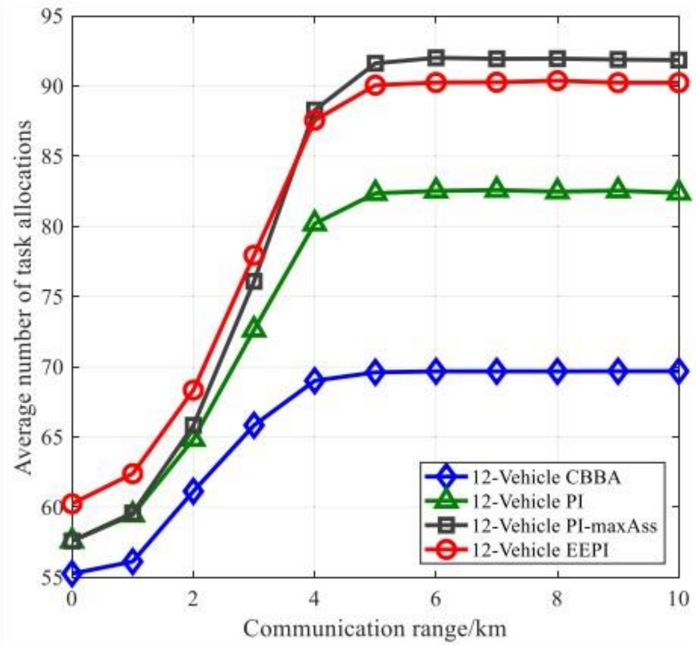

Figure 10. Task allocation with numerous communication ranges

D. Perception Range

The experimental evaluation demonstrated that the EEPI algorithm exhibited superior performance compared to both the CBBA and PI algorithms. Specifically, as the perception range was expanded, there was a notable increase in the task allocations achieved by the EEPI algorithm. The superiority of EEPI over PI-maxAss was found to be more pronounced when larger perception ranges were considered.

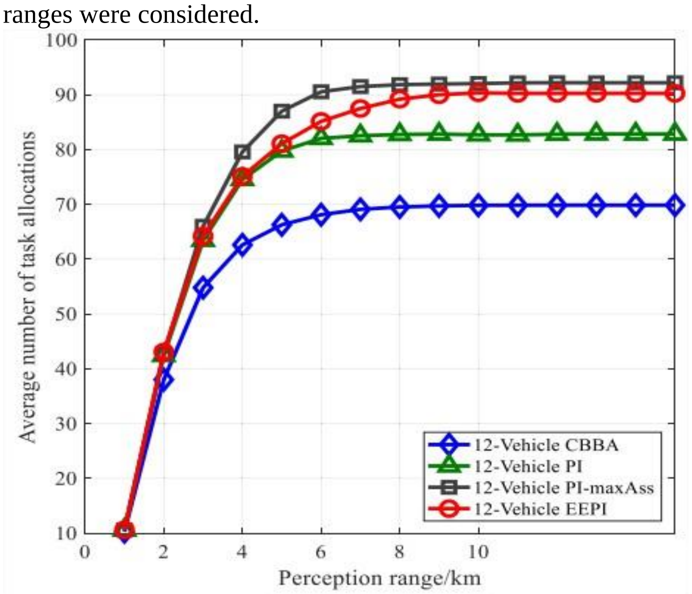

Figure 11. Task allocation with numerous perception ranges.

V. R&D ON HARDWARE-IN-THE-LOOP

In order to enhance the credibility and reliability of the proposed Energy Efficient Path Planning Algorithm (EEPI), a series of hardware-in-the-loop experiments were undertaken. These experiments were specifically designed to validate the effectiveness and efficiency of the EEPI algorithm on a platform that utilised unmanned aerial vehicles (UAVs). The experimental configuration, as illustrated in Figure 12, encompassed tangible onboard modules, namely an onboard processing unit, a flight controller, and an Adhoc network. The experimental setup involved the definition and display of scenarios, unmanned aerial vehicles (UAVs), and tasks on a designated host machine. Subsequently, real onboard modules were employed during the execution of the experiments.

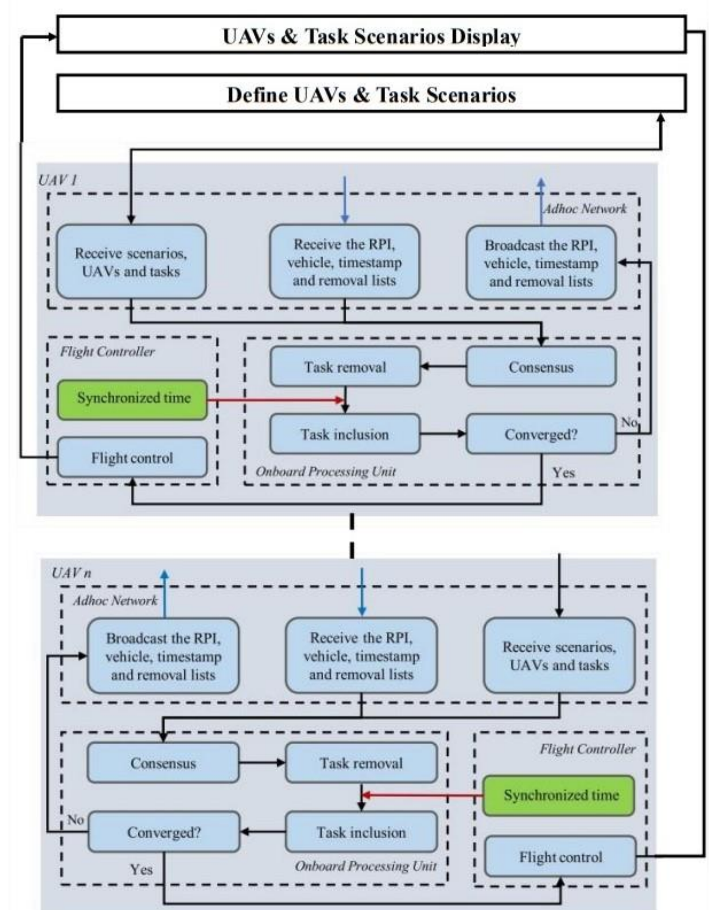

Figure 12. Main strategy for running distributed task allocation algorithms on experimental hardware-in-the-loop system.

Figure 13 depicts the tangible components integrated within each Unmanned Aerial Vehicle (UAV), specifically highlighting the presence of the onboard processing unit, flight controller, and Adhoc network. In order to implement distributed task allocation algorithms, the host machine transmitted configurations to the respective unmanned aerial vehicles (UAVs). The conducted experiments encompassed the utilisation of Unmanned Aerial Vehicles (UAVs) in their trajectory towards a designated mission area. Subsequently, task allocation procedures were initiated, followed by the execution of tasks in accordance with the computed trajectories. The experimental setup consisted of a rectangular mission area, involving three unmanned aerial vehicles (UAVs) and a total of eighteen tasks. In stark contrast to simulations, the positions of fixed-wing unmanned aerial vehicles (UAVs) exhibited a dynamic nature. Task activation occurred upon the proximity of an Unmanned Aerial Vehicle (UAV) to its designated target, within a predetermined threshold distance.

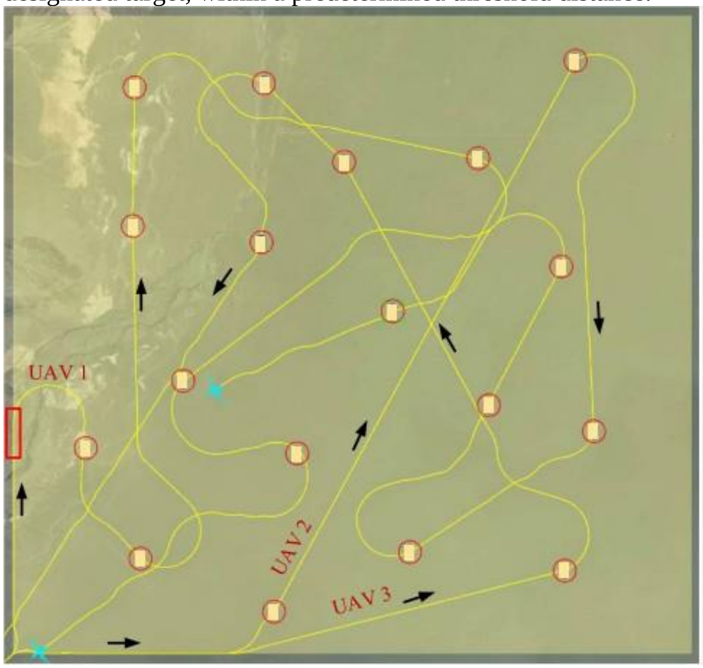

Figure 13. Each UAV Really Has Its Own Set of Onboard Modules and test scenarios.

E. Experimental Results

A series of ten experiments were undertaken, wherein different combinations of deadlines, task durations, and UAV active times were examined. Table II provides a summary of the task allocations and convergence time for the Cooperative Bundle Algorithm (CBBA), the Priority Index (PI) method, the PI-maxAss approach, and the proposed Enhanced Efficiency Priority Index (EEPI).

Table 2. Priority of Tasks and Convergence Timing in Hardware-in-the-Loop Studies

No.	Task allocations				Time to converge			
	CBBA	PI	PI-maxAss	EEPI	CBBA	PI	PI-maxAss	EEPI
1	12	14	16	16	11.4	10.7	18.7	8.0
2	12	15	16	16	12.0	11.1	20.5	7.5
3	13	16	16	16	9.8	16.0	22.9	6.6
4	13	15	16	16	10.2	11.9	18.0	7.6
5	12	13	15	15	10.1	13.8	22.0	9.6
6	15	15	16	16	10.5	10.0	16.2	6.0
7	15	15	16	16	9.4	13.7	21.7	12.0
8	13	15	16	16	8.4	17.2	22.0	6.3
9	13	15	16	15	9.5	12.2	20.0	7.1
10	13	14	16	16	9.1	10.8	18.8	7.1

The experimental results consistently showed that the proposed Enhanced Efficient Path Iteration (EEPI) algorithm exhibited superior task allocations when compared to the CBBA and PI algorithms. In fact, in the majority of scenarios, the EEPI algorithm performed on par with the PI-maxAss algorithm. The convergence time observed with the proposed Energy-Efficient Power Iteration (EEPI) algorithm consistently exhibited the lowest values, frequently amounting to less than half of the convergence time achieved by the PI-maxAss algorithm. It is worth noting that, in comparison to simulations, the convergence time of the proposed Energy-Efficient Pathfinding Algorithm (EEPI) occasionally exhibited a shorter duration than the Consensus-Based Bundle Algorithm (CBBA). This discrepancy can be attributed to hardware limitations encountered when employing a smaller fleet of Unmanned Aerial Vehicles (UAVs). The conducted experiments demonstrated that, within a hardware-in-the-loop setting, the efficiency of the proposed Enhanced Evolutionary Particle Intelligence (EEPI) algorithm surpassed that of alternative algorithms in terms of both task allocations and convergence time.

VI. CONCLUSION

An innovative and highly efficient distributed task allocation method is presented in this paper. This method is referred to as the Effective and Efficient Task Allocation (EEPI), and it is a revolutionary approach to job allocation. The fundamental goal of EEPI is to maximise the total number of jobs that are successfully completed while minimising the amount of time that must be spent rescheduling. When conducting this study, the researchers took into consideration the initiation times, work deadlines, and fuel limits that are associated with automobiles. In order to facilitate the allocation of extra jobs while simultaneously attaining convergence in a decreased number of iterations, a novel cost function was developed inside the implementation of the EEPI framework. There is a mechanism that has been established in order to handle the problem of invalid Relative Priority Indices (RPIs) and Inter-Task Priority Indices (IPIs). This mechanism is responsible for releasing all of the tasks that are contained within a vehicle's task list in the case that it eliminates the highest number of tasks during the phase that is dedicated to task removal. The execution of this particular strategy has demonstrated a significant level of effectiveness in increasing the total number of tasks that have been successfully completed thanks to the deployment of this particular strategy.

In order to demonstrate the effectiveness and efficiency of the proposed EEPI, a complete analysis was conducted, which included extensive simulations and experiments incorporating hardware-in-the-loop. Our primary purpose is to examine the relationship between the number of iterations and the various network topologies that we will be conducting in the course of our upcoming research efforts. In addition, the purpose of our research is to address the difficulties that are associated with the assignment of duties in situations that include the cooperation of several vehicles in the execution of a task simultaneously, as well as in circumstances in which a single vehicle is responsible for executing multiple jobs simultaneously.

.

REFERENCES

[1] N. R. Gans and J. G. Rogers, "Cooperative multirobot systems for military applications," Current Robot. Rep., vol. 2, no. 1, pp. 105–111,Mar. 2021.

[2] B. Huang, Y. Yang, Y. Tsai, and G. Yang, "A reconfigurable multirobot cooperation workcell for personalized manufacturing," IEEE Trans. Autom. Sci. Eng., vol. 19, no. 3, pp. 2581–2590, Jul. 2022.

[3] P. Huang, L. Zeng, X. Chen, K. Luo, Z. Zhou, and S. Yu, "Edge robotics: Edge-computing-accelerated multirobot simultaneous localization and mapping," IEEE Internet Things J., vol. 9, no. 15, pp. 14087–14102, Aug. 2022.

[4] A. Khan, B. Rinner, and A. Cavallaro, "Cooperative robots to observe moving targets: Review," IEEE Trans. Cybern., vol. 48, no. 1, pp. 187–198, Jan. 2018.

[5] A. Quattrini Li, R. Cipolleschi, M. Giusto, and F. Amigoni, "A semantically-informed multirobot system for exploration of relevant areas in search and rescue settings," Auto. Robots, vol. 40, no. 4, pp. 581–597, Apr. 2016.

[6] F. Amigoni, J. Banfi, and N. Basilico, "Multirobot exploration of communication-restricted environments: A survey," IEEE Intell. Syst., vol. 32, no. 6, pp. 48–57, Nov. 2017.

[7] Y. Liu and G. Nejat, "Multirobot cooperative learning for semiautonomous control in urban search and rescue applications," J. Field Robot., vol. 33, no. 4, pp. 512–536, Jun. 2016.

[8] D. S. Drew, "Multi-agent systems for search and rescue applications," Current Robot. Rep., vol. 2, no. 2, pp. 189–200, Mar. 2021.

[9] A. Macwan, J. Vilela, G. Nejat, and B. Benhabib, "A multirobot path planning strategy for autonomous wilderness search and rescue," IEEE Trans. Cybern., vol. 45, no. 9, pp. 1784–1797, Sep. 2015.

[10] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging UAVs for disaster management," IEEE Pervasive Comput., vol. 16, no. 1, pp. 24–32, Jan. 2017.

[11] S. Shriyam and S. K. Gupta, "Incorporation of contingency tasks in task allocation for multirobot teams," IEEE Trans. Autom. Sci. Eng., vol. 17, no. 2, pp. 809–822, Apr. 2020.

[12] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," IEEE Trans. Ind. Informat., vol. 9, no. 1, pp. 427–438, Feb. 2013.

[13] Z. Chen, J. Alonso-Mora, X. Bai, D. D. Harabor, and P. J. Stuckey, "Integrated task assignment and path planning for capacitated multiagent pickup and delivery," IEEE Robot. Autom. Lett., vol. 6, no. 3,pp. 5816–5823, Jul. 2021.

[14] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot task allocation: A review of the state-of-the-art," in Cooperative Robots and Sensor Networks, vol. 604. 2015, pp. 31–51.

[15] J. Chen and D. Sun, "Coalition-based approach to task allocation of multiple robots with resource constraints," IEEE Trans. Autom. Sci. Eng., vol. 9, no. 3, pp. 516–528, Jul. 2012.