applied sciences

MDPI

Article

Centralized Mission Planning for Multiple Robots Minimizing Total Mission Completion Time

Nam Eung Hwang *, Hyung Jun Kim and Jae Gwan Kim

Hanwha Systems Co., Seongnam-si 13524, Gyeonggi-do, Republic of Korea
* Correspondence: skadnd144@hanwha.com

Abstract: Most mission planning algorithms solve multi-robot-multi-mission problems based on mixed integer linear programming. In these algorithms, the rewards (or costs) of missions for each robot are calculated according to the purpose of the user. Then, the (robot-mission) pair that has maximum rewards (or minimum costs) is found in the rewards (or costs) table and the mission is allocated to the robot. However, it is hard to design the reward for minimizing total mission completion time because not only a robot, but also the whole robots' mission plans must be considered to achieve the purpose. In this paper, we propose centralized mission planning for multi-robot-multi-mission problems, minimizing total mission completion time. First, mission planning for single-robot-multi-mission problems is proposed because it is easy to solve. Then, this method is applied for multi-robot-multi-mission problems, adding a mission-plan-adjustment step. To show the excellent performance of the suggested algorithm in diverse situations, we demonstrate simulations for 3 representative cases: a simple case, which is composed of 3 robots and 8 missions, a medium case, which is composed of 4 robots and 30 missions, and a huge case, which is composed of 6 robots and 50 missions. The total mission completion time of the proposed algorithm for each case is lower than the results of the existing algorithm.

Keywords: mission planning; task allocation; task alignment; task planning; multi-robot systems; automation; robot control

check for updates

Citation: Hwang, N.E.; Kim, H.J.; Kim, J.G. Centralized Mission Planning for Multiple Robots Minimizing Total Mission Completion Time. *Appl. Sci.* **2023**, *13*, 3737. https://doi.org/10.3390/app13063737

Academic Editor: Diego González-Aguilera

Received: 21 February 2023
Revised: 10 March 2023
Accepted: 13 March 2023
Published: 15 March 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

As automation technology highly advances, concepts of swarming, Manned and Unmanned Teaming (MUM-T), etc., occurred [1–9]. These concepts focused on overcoming a variety of missions which are hard for single robot by using multiple robots [10–14]. For example, multiple robots can complete surveillance and reconnaissance missions of a specific area within a short time, while a single robot takes much more time. If there are multiple missions to solve, making a mission plan for each robot considering the purpose of the user is important for efficiency. Although there are various purposes, such as conducting important missions first, minimizing fuel consumption (or minimizing total moving distances), etc., most users want to complete whole missions as soon as possible, which means minimizing total mission completion time.

Mission planning technologies have been developed in various ways [15–33]. One of the general mission planning methods is Consensus-based Bundle Algorithm (CBBA) which is based on scores, i.e., rewards or costs of missions for each robot [25]. The CBBA algorithm is separated into two steps: the bundle construction step and conflict resolution step. In the bundle construction step, each robot calculates the sequence of missions based on the scores under the assumption that the robot accomplishes all missions. The scoring scheme is designed using the expected start/finish time of tasks, distances between robots and missions, etc., based on the user's purpose. In the conflict resolution step, each robot decides which missions it will do based on scores calculated in the bundle construction step. Usually, the robot who has the highest score will take the mission that has the maximum

score. CBBA is a deterministic allocation algorithm, so results are always same if the inputs, the information of the robots and missions, are same. In addition, the computation cost of CBBA is much lower than for other heuristic algorithms.

The modified task allocation that can handle some constraints based on CBBA is Coupled-Constraint CBBA (CCBBA) [26]. The missions that have a constraint relationship are grouped as 'Activity' and each mission in the activity is set as 'Element'. Such settings enable easy judgement of whether a mission can be allocated to a robot or not. In addition, it uses optimistic and pessimistic bidding strategies. Each robot bids to the missions based on the strategy for obeying the constraints. However, there are some problems regarding convergence and calculation time for mission planning. Hence, our previous work proposed the Task Allocation and Alignment based on Constraint Table and Alignment Rules (TACTAR), which is modified based on CBBA for centralized mission planning systems [33]. It handles some spatial and temporal constraints using a constraint table and four alignment rules for mission planning. These methods commonly use only the robot's own information to score the missions, but the plans of all robots should be considered to minimize the total task completion time. Hence, CBBA, CCBBA, and TACTAR cannot be used for mission allocation, minimizing total mission completion time without any modification.

Qin et al. proposed path planning algorithms for multiple Unmanned Aerial Vehicles (UAVs) to minimize mission completion time [24]. The problem in this paper is defined as a collection of sparse multiple sensors. The sensor points which have distances less than the communication distance of UAVs are grouped, and the center point of each group is defined as hovering point. Then, an ant colony algorithm for solving Traveling Salesman Problem (TSP) is used to visit whole hovering points by one UAV, and a cycle is made as the result. The k-cycles algorithm is used for decomposing the cycle into the number of UAVs. However, this algorithm is based on basic TSP, which means that UAVs must come back to home, and the home positions for all UAVs must be the same. In addition, the ant colony algorithm is used for solving TSP, so the calculation cost of the algorithm will be fatal when the number of hovering points becomes larger.

Wang et al. also proposed path planning algorithms for multiple fixed-wing UAVs for minimizing mission completion time [34]. The algorithm focused on the path planning of fixed-wing UAVs for UAV-to-UAV (U2U) communication with the aim of minimum information transmission time in the presence of ground transmitters. The constraints contain the communication throughput requirement, interference from ground transmitters, the speed and acceleration range of each UAV, and the minimum communication distance of UAVs. However, they only consider only one mission: communicating two UAVs with each other. Hence, the algorithm calculates the local path planning for each UAV to minimize information transmission time, not the sequence of missions for each UAV.

In this paper, we propose a centralized mission planning algorithm for multi-robot-multi-mission problems, minimizing total mission completion time. The proposed algorithm is designed based on CBBA to overcome some limitations such as computation time and necessity to set the same home for all robots in the above algorithms. First, we introduced the algorithm for single-robot-multi-mission problems, minimizing total mission completion time based on the method using the bundle construction step in CBBA. It focused on how to decide the sequence of missions that the robot has to work on. Then, the algorithm was expanded for multi-robot-multi-mission problems. To overcome limitations of CBBA, CCBBA, and TACTAR, we proposed bundle adjustment, which adjusts the mission plan only if the total mission completion time could be lower by the adjustment. In order to apply the status of each robot in real time, the proposed algorithm should be run iteratively. However, if the positions of robots are changed, the mission plan could fall into another local optimal solution, so the mission plan could be changed frequently. To prevent this phenomenon, the initial bundle construction, which is based on the arrival times to missions for robots, is performed only at the first run, and the previous mission plan is used from the second iteration instead of the initial bundle construction. Then, some simulation results using the proposed algorithm will be provided.

2. Preliminaries

The mission planning problem for minimizing total mission completion time is to find a sequence of missions for each robot that has the minimum total mission completion time as seen below:

$$\mathrm{Min}\big(\mathrm{Max}\big(t_e(j,\ p_i)x_{ij}\big)\big) \tag{1}$$

subject to

$$\sum_{i=1}^{N_r} x_{ij} \leq 1,\ \forall j \in J \tag{2}$$

$$\sum_{j=1}^{N_m} x_{ij} \leq L_m,\ \forall i \in I \tag{3}$$

$$\sum_{i=1}^{N_r}\sum_{j=1}^{N_m} x_{ij} = N_{min} \tag{4}$$

$$x_{ij} \in \{0,\ 1\} \tag{5}$$

where $t_e(j,\ p_i)$ is finish time of mission j if robot i conduct mission j along the path p_i; x_{ij} is the flag that is 1 if mission j is taken to robot i and 0, or otherwise, N_r and N_m are the number of robots and missions, respectively, L_m is the maximum number of missions that the robot can take, and N_{min} is the minimum number between N_m and $N_r L_m$. I and J are robot and task group, respectively, so the number of elements in I is N_r and the number of elements in J is N_m. Note that one mission is taken to only one robot as described in Equation (2).

The mission planning is composed of two parts: mission allocation and mission alignment. The mission allocation is to decide which robot will take which mission. The mission alignment is to calculate the start time and the finish time of each mission based on the result of mission allocation. Some mission planning algorithms conduct mission allocation at first for all missions; then, the mission alignment is performed once based on the mission allocation result. The other mission planning algorithms conduct mission allocation for one mission, and then the mission alignment is performed for one mission. This procedure repeats for all missions. This sequence is used when the mission allocation uses the start time or the finish time of each mission for the results of mission alignment.

3. Related Works

This section reviews the most common mission planning works related to the proposed algorithm; one is CBBA and the other is TACTAR [25,33].

3.1. Consensus-Based Bundle Algorithm (CBBA)

CBBA is one of the mission planning algorithms for the multi-robot-multi-mission problem. It is inspired by a decentralized auction process. CBBA is composed of two steps: the bundle construction step and the conflict resolution phase. In bundle construction step, robots make their mission lists, called a bundle, and decide their own sequence of missions assuming that each robot takes all missions. For this step, each agent calculates the rewards of the possible mission. The reward calculation scheme depends on the purpose of users. The procedure of bundle construction is as in Algorithm 1.

In lines 1–4, the results of the previous iteration are copied into the results of the current iteration. $\boldsymbol{y}_i$, $\boldsymbol{z}_i$, $\boldsymbol{b}_i$, and $\boldsymbol{p}_i$ are winning bids (the highest rewards of each mission) that robot i knows, winning robots (who take each mission) that robot i knows, the bundle (which mission that robot to do) of robot i, and the sequence of the missions of the robot i, respectively. Note that t is the number of iterations (maybe L_m or N_m times), not time. If the number of bundles for robot i is lower than L_m, the rewards of missions that are not included in the bundle is calculated. $S_i^{\boldsymbol{p}_i}$ means the total reward of robot i if it completes along the sequence $\boldsymbol{p}_i$. $\boldsymbol{p}_i(+)_n\{j\}$ means adding mission j into the nth position of sequence

$\boldsymbol{p}_i$, so the reward of the mission is the highest score difference $\boldsymbol{S}_i^{\boldsymbol{p}_i(+)_n\{j\}} - \boldsymbol{S}_i^{\boldsymbol{p}_i}$ when the mission is added to the sequence. In line 7, the reward of the mission is compared to the winning bid of the mission. $\boldsymbol{h}_{ij}$ is 1 if the reward is higher than the winning bid, and 0 otherwise. If the rewards of all possible missions are calculated, then the robot i chooses the mission that has the highest rewards. Finally, the chosen mission is added to the results of the current iteration.

Algorithm 1 CBBA Phase 1: Bundle construction

1: $\boldsymbol{y}_i(t) = \boldsymbol{y}_i(t-1)$
2: $\boldsymbol{z}_i(t) = \boldsymbol{z}_i(t-1)$
3: $\boldsymbol{b}_i(t) = \boldsymbol{b}_i(t-1)$
4: $\boldsymbol{p}_i(t) = \boldsymbol{p}_i(t-1)$
5: **while** $|\boldsymbol{b}_i| < L_t$, **do**
6: $c_{ij} = max_{n \leq |\boldsymbol{p}_i|} S_i^{\boldsymbol{p}_i(+)_n\{j\}} - S_i^{\boldsymbol{p}_i}$, $\forall j \in \boldsymbol{J} \setminus \boldsymbol{b}_i$
7: $h_{ij} = \mathrm{II}\left(c_{ij} > y_{ij}\right)$, $\forall j \in \boldsymbol{J}$
8: $JJ_i = argmax_j c_{ij} \cdot h_{ij}$
9: $n_{i,JJ_i} = argmax_n \; S_i^{\boldsymbol{p}_i(+)_n\{JJ_i\}}$
10: $\boldsymbol{b}_i = \boldsymbol{b}_i(+)_{end}\{JJ_i\}$
11: $\boldsymbol{p}_i = \boldsymbol{p}_i(+)_{n_{i,JJ_i}}\{JJ_i\}$
12: $y_{i,JJ_i}(t) = c_{i,JJ_i}$
13: $z_{i,JJ_i}(t) = i$
14: **end while**

The next step of CBBA is the conflict resolution step, which is making a decision on who will take the mission by comparing the rewards of each robot. Basically, CBBA is for a decentralized mission planning system, so two communicable robots decide the owner of missions following the rules described in Table 1.

Table 1. Conflict resolution rules for mission j of robot i which communicates with robot k.

Robot k's (Sender) z_{kj} Is	Robot k's (Sender) z_{kj} Is	Receiver's Action (Default: Leave)
k	i	update if $(y_{kj} > y_{ij})$
	k	update
	$m \notin \{i, k\}$	update if $(s_{km} > s_{im})$ || $(y_{kj} > y_{ij})$
	none	update
i	i	leave
	k	reset
	$m \notin \{i, k\}$	reset if $(s_{km} > s_{im})$
	none	leave
$m \notin \{i, k\}$	i	update if $(s_{km} > s_{im})$ & $(y_{kj} > y_{ij})$
	k	update if $(s_{km} > s_{im})$, reset else
	m	update if $(s_{km} > s_{im})$
	$n \notin \{i, k, m\}$	update if $(s_{km} > s_{im})$ & $((s_{kn} > s_{in})$ || $(y_{kj} > y_{ij}))$
		reset if $(s_{kn} > s_{in})$ & $(s_{im} > s_{km})$
	none	update if $(s_{km} > s_{im})$
none	i	leave
	k	update
	$m \notin \{i, k\}$	update if $(s_{km} > s_{im})$
	none	leave

If CBBA is used in centralized mission planning, the conflict resolution step based on the rules in Table 1 is replaced into the Sequential Greedy Algorithm (SGA). When the bundle construction step is performed, the rewards of missions for each robot are calculated

and the central computer makes a reward table based on the rewards of each robot. Then, the SGA just finds the (robot-mission) pair that has the maximum reward and conducts mission allocation and alignment.

3.2. Centralized Task Allocation and Alignment Based on Constraint Table and Alignment Rules (TACTAR)

CBBA cannot solve the problem with several constraints. The constraints are separated into two types: temporal and spatial constraints. The temporal constraints are constraints about the start/finish time of missions. For instance, 'mission A ends before mission B starts', 'mission A starts simultaneously with mission B', etc., are temporal constraints. The spatial constraints are constraints about mission allocation. For example, 'mission A is local MUTEX with mission B' means mission A cannot be allocated to the robot who takes mission B and vice versa. TACTAR is a centralized mission planning algorithm that can handle spatial and temporal constraints based on CBBA. Supported constraints in TACTAR are depicted in Table 2.

Table 2. Supported type of constraints in TACTAR.

Name	Description
Simultaneous	Mission A and B must start at the same time
After	Mission A must start after mission B's finish time
Start During	Mission A must start between mission B's start time and finish time
End During	Mission A must end between mission B's start time and finish time
Local MUTEX	Mission A and B are locally mutually exclusive
Global MUTEX	Mission A and B are globally mutually exclusive

The purpose of TACTAR is to calculate the mission plan considering spatial and temporal constraints quickly. TACTAR does not do as described in line 6 of Algorithm 1, just calculate rewards of missions, assuming missions are performed right after the robot's last mission. The optimization performance may suffer slightly, but calculation speed is boosted much faster than CBBA. In addition, some filtering process is added to TACTAR before the beginning of the mission allocation and alignment to not violate constraints. In the mission alignment of TACTAR, some delay or renewal of the mission time (arrival time to mission area, start/finish time of mission) are performed based on constraint table and alignment rules. Further details can be found in [33].

4. Mission Planning Scheme

In this section, we propose a centralized mission planning algorithm for minimizing the total mission completion time. It is based on minimizing total mission completion time for a single-robot-multi-mission problem, modified from the bundle construction in CBBA. So, the algorithm for single-robot is introduced first, and it expands to the algorithm for the multi-robot problems. Note that only mission alignment is needed in single-robot-single-mission problems. In addition, note that this section is focused on the mission planning, not path planning, for minimizing total mission completion time.

4.1. Single-Robot-Multi-Mission Problem

In Single-Robot-Multi-Mission (SRMM) problem, one robot has to conduct all missions, so there is no need to select missions to conduct. All the mission planning has to achieve is to decide the sequence of missions and to calculate start/finish time of each mission. To find the optimal sequence of missions, we propose a process similar to line 6 of Algorithm 1 as described in Algorithm 2.

Algorithm 2 Mission planning: SRMM

1: **while** $|\boldsymbol{p}_i| < L_m$, **do**
2: $t_{ij} = min_{n \leq |\boldsymbol{p}_i|} t_i^e(\boldsymbol{p}_i(+)_n\{j\}), \forall j \in \boldsymbol{J} \backslash \boldsymbol{p}_i$
3: $JJ_i = argmin_j\left(t_{ij}\right)$
4: $n_{i,\,JJ_i} = argmin_n\left(t_i^e(\boldsymbol{p}_i(+)_n\{JJ_i\})\right)$
5: $\boldsymbol{p}_i = \boldsymbol{p}_i(+)_{n_{i,\,JJ_i}}\{JJ_i\}$
6: **end while**

In line 2, the total mission completion time is calculated when a mission is added to nth space of sequence using the speed of the robot, the distances between the robot and mission areas, and duration of missions. It repeats for all missions and lengths of sequence. Then, it finds the (mission-space) pair that has the minimum total mission completion time. Finally, the found mission is added to the found space of sequence. Lines 2–5 of Algorithm 2 repeat until all missions (or L_m missions) are allocated. After the final mission is allocated, the start/finish times of each mission are calculated.

4.2. Multi-Robot-Multi-Mission Problem

Unlike the SRMM problems, each robot must select which missions it will perform in Multi-Robot-Multi-Mission (MRMM) problems, considering the equipment status of robots. For example, the robot that does not have camera cannot conduct surveillance and reconnaissance missions. However, it is hard to conduct a mission allocation because not only a robot but also the whole robots' planning results must be considered to achieve total mission completion time minimization. So, we propose initial bundle construction and bundle adjustment methods as described in Algorithm 3.

Algorithm 3 Mission planning: MRMM

1: **if** first time
2: *Construct initial bundle*
3: **else**
4: bundle = result of adjustment
5: **end if**
6: *Adjust bundle*

If it is first time for the mission planning, initial bundle construction step runs. If not, the result of the previous adjustment is copied into the bundle for preventing to fall into local optimization solutions during mission planning iterations. Then, the bundle adjustment for total mission completion time minimization is run.

In initial bundle construction, as described in Algorithm 4, each robot calculates the expected finish time of missions based on the positions of the robot and missions, the speed of the robot, and the duration of each mission for the robot. Then, the table of finish time of missions for robots will be made. The (robot-mission) pair that has the minimum finish time of the mission is found in the table. If the pair has an infinity finish time of mission (initial value of the table), all possible missions are already added to the bundles, so it breaks out of the iterations. If not, it checks the selected robot that can conduct the selected mission. If the robot cannot conduct the mission, $t_{i_{sel}j_{sel}}$ in the table is set to infinity (or initial value) and another pair in the table is found. If the robot can conduct the mission, the mission is added to the bundle of the robot. The column (or row) about the mission is set to infinity (or initial value) in the table, and the procedure is repeated until all missions are added to the bundle.

Algorithm 4 Construct initial bundle

$t_{ij} = d_{ij}/v_i + dur_{ij}, \forall i \in \boldsymbol{I}, \forall j \in \boldsymbol{J}$
while 1
  $i_{sel} = argmin_i\left(t_{ij}\right)$
  $j_{sel} = argmin_j\left(t_{i_{sel}j}\right)$
  if $t_{i_{sel}j_{sel}} = \infty$
    break
  else
    if i_{sel} cannot do j_{sel}
      $t_{i_{sel}j_{sel}} = \infty$
    else
      $\boldsymbol{b}_{i_{selelct}} = \boldsymbol{b}_{i_{sel}}(+)_{end}\{j_{sel}\}$
      $\boldsymbol{t}_{j_{sel}} = \infty$
    end if
  end if
end while

In the bundle adjustment, line 6 of Algorithm 3, the robot that has the maximum total mission completion time is found. Then, one of the missions in the bundle of that robot is moved to the bundles of the others who can conduct the mission. The total mission completion time for each robot can be calculated based on the bundles following Algorithm 2, because fixing the bundles of each robot means MRMM problems are simplified into SRMM problems for each robot. The $(i_s,\ j_s)$ pair that has the minimum total mission completion time is searched. If there are multiple $(i_s,\ j_s)$ pairs that have the same minimum total completion time, a $(i_s,\ j_s)$ pair in which the larger value is the smallest among the total mission completion time of i_{max} and i_s is selected. Finally, the total mission completion times for the results of previous and current iterations are compared. If the current one is lower than the previous one, the current bundle and total mission completion time are copied into the previous results. Then, it goes to line 4 of Algorithm 5 and the procedures are repeated. If not, it means the previous results are the optimal mission plans for minimizing the total mission completion time, so it outputs the previous results.

Algorithm 5 Adjust bundle

$t^e_{max,prev} = max(\boldsymbol{t}^e)$
$\boldsymbol{b}_{prev} = \boldsymbol{b}$
while 1
  $i_{max} = argmax_i\left(t^e_i\right)$
  for $n = 1 : size\left(\boldsymbol{b}_{i_{max}}\right)$
    $\boldsymbol{b}_{i_{max}} = \boldsymbol{b}_{i_{max},prev}$
    $j_{sel}(n) = \boldsymbol{b}_{i_{max}}(n)$
    $\boldsymbol{b}_{i_{max}} = \boldsymbol{b}_{i_{max}}(-)_n$
    $\boldsymbol{b}_i = \boldsymbol{b}_i(+)_{end}\{j_{sel}(n)\}, \forall i \in \boldsymbol{I}\backslash i_{max}$ if i can do $j_{sel}(n)$
    calculate $max\left(\boldsymbol{t}^e_{in}\right)$ for $\boldsymbol{b}$
  end for
  $t^e_{max} = min\left(max\left(\boldsymbol{t}^e_{in}\right)\right)$
  $i_s = argmin_i\left(max\left(\boldsymbol{t}^e_{in}\right)\right), \forall i \in \boldsymbol{I}\backslash i_{max}$
  $j_s = j_{sel}\left(argmin_n\left(max\left(\boldsymbol{t}^e_{in}\right)\right)\right)$
  if multiple $(i_s,\ j_s)$ that has same t^e_{max}
    choose that has $min\left(\max\left(t^e_{i_{max}},\ t^e_{i_s}\right)\right)$
  end if
  if $t^e_{max,prev} > t^e_{max}$
    $t^e_{max,prev} = t^e_{max}$
    $\boldsymbol{p}_{prev} = \boldsymbol{p}$
  else
    break
  end if
end while

5. Result and Discussion

In this section, some simulation results are shown to assess the algorithm that we have proposed. We simulated three cases, a simple case, medium case, and huge case, to prove the performance of proposed algorithm. The simple case is composed of 3 robots and 8 missions, the medium case is composed of 4 robots and 30 missions, and the huge case is composed of 6 robots and 50 missions. The speed of all robots and the duration of all missions are equally set to 2 m/s and 5 s, respectively, to simplify the problem in the simple case. In the medium and huge cases, the speed of each robot and the duration time of each mission are set differently to represent the general situations. The detailed scenarios of simulations are described in Table 3.

Table 3. The scenarios of simulations.

Contents	Specification		
	Simple	Medium	Huge
# of robots	3	4	6
# of missions	8	30	50
Pos. of robots	[1, 0], [2, 0], [3, 0]	[1, 0], [1, 1], [3, 0], [3, 1]	[1, 0], [1, 1] [2, 0], [2, 1] [3, 0], [3, 1]
Pos. of missions	[−2, 3], [−4, 3] [−4, 5.1], [−2, 5.2] [7, 7], [9, 7] [9, 9.1], [7, 9.2]	20·rand(30, 2)-10	20·rand(50, 2)-10
Spd. of robots	2 [m/s] for all	2.0 [m/s], 1.5 [m/s], 1.0 [m/s], 0.5 [m/s]	2.0 [m/s], 1.8 [m/s], 1.6 [m/s], 1.4 [m/s], 1.2 [m/s], 1.0 [m/s]
Dur. of missions	5 [s] for all	2.1:0.1:5 [s]	2.1:0.1:7 [s]

CBBA and TACTAR are used to show that the proposed algorithm is improved over the existing algorithm. The total mission completion time and computational cost are set as performance indicators. We used OpenMP parallel computing to run each algorithm, and the computational cost of each algorithm is measured based on the result of the parallel computation. The environment of simulations is described in Table 4.

Table 4. The environment of simulations.

Contents		Specification
CPU		Intel® Core™ i5-10500 3.10 GHz (6 cores, 12 threads)
GPU		Intel® UHD Graphics 630
RAM		16 GB
Storage		256 GB SSD
Program	Calculation	Visual Studio Professional 2013 C++
	Plotting	MATLAB 2018b

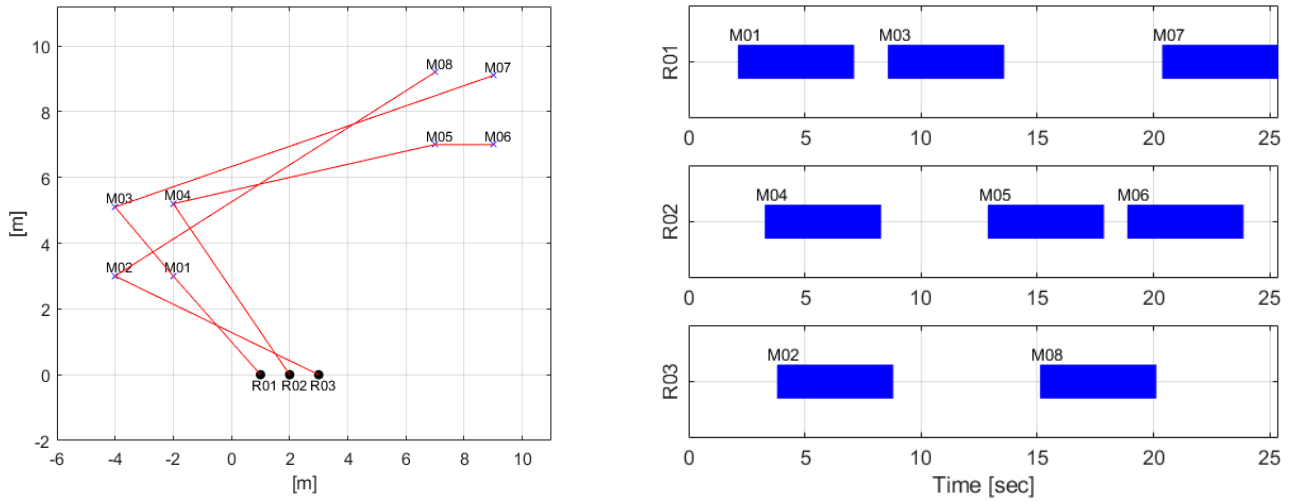
5.1. The Simple Case

The results of the simple case are shown in Figures 1–3. Figure 1 depicts the results of CBBA, Figure 2 depicts the results of TACTAR, and Figure 3 depicts the results of the proposed algorithm. Panel (a) in Figures 1–3 shows the 2D situation map to easily check the mission plan result of each algorithm. The 'R' and 'M' means robot and mission, respectively. The following number means the index of robots and missions. The red lines mean the sequence of missions for each robot. For example, R01 will conduct M01, M03, and M07 sequentially, as seen in Figure 1a. As seen in Figures 1 and 2, the mission plan result of CBBA is absolutely the same as with the result of TACTAR. However, the mission plan result of the proposed algorithm is totally different to them. R01 and R03 will conduct

(M01-M02-M03) and (M05-M06-M07), which are the closest missions to the R01 and R03, respectively. R02 will conduct M04 and M08, which were the last missions for R01 and R03, so the total mission completion time is minimized.

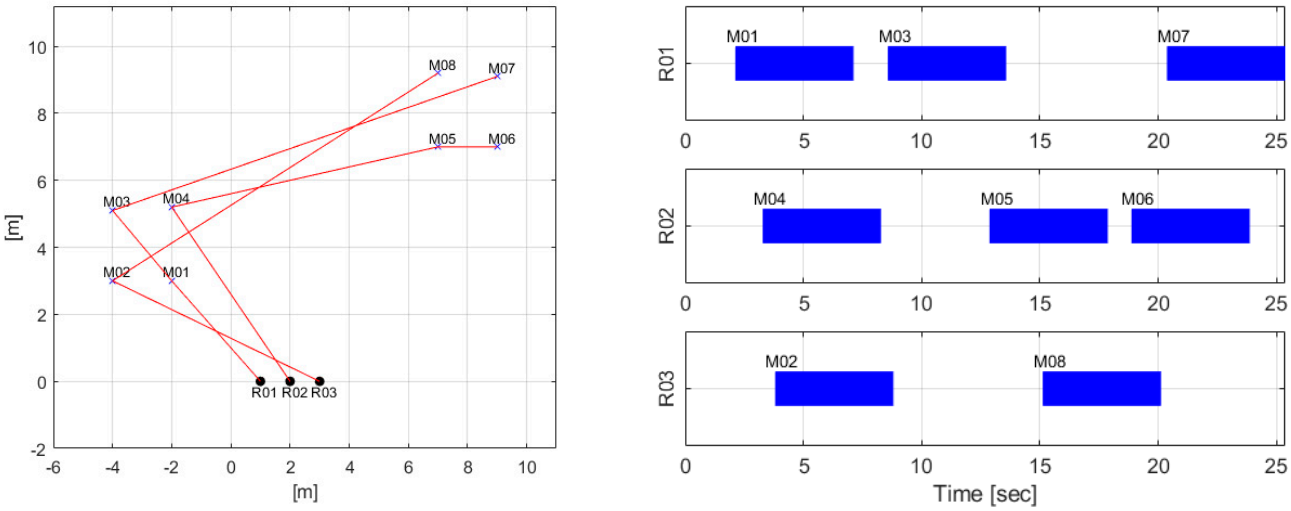
Panel (b) in Figures 1–3 shows a timetable of each algorithm to easily check the mission completion time of each robot. The blue boxes mean the duration times of missions. The 'R', 'M', and the number after them have the same meanings as in the panel (a). In the panel (b) of Figures 1 and 2, R01 will complete the last mission, M07, at 25.37 s. So, the total mission completion times of CBBA and TACTAR are 25.37 s. In the panel (b) of Figure 3, R03 will complete the last mission, M07, at 21.08 s, which is about 4.3 s lower than the results of CBBA and TACTAR.

The calculation times of CBBA, TACTAR, and the proposed algorithm for the simple case are 0.0055 s, 0.0068 s, and 0.0155 s, respectively. The computational cost of the proposed algorithm is about 2~3 times larger than the others, but it is still fast enough to run it in real-time.



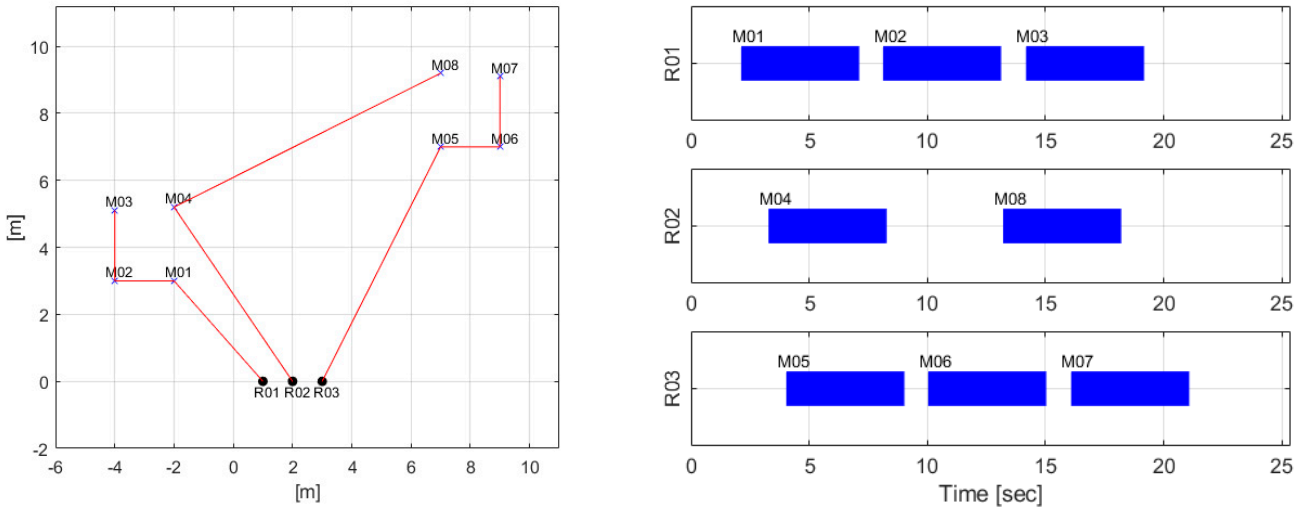
(**a**) (**b**)

Figure 1. The results for simple case using CBBA: (**a**) 2D situation map; (**b**) timetable.



(**a**) (**b**)

Figure 2. The results for simple case using TACTAR: (**a**) 2D situation map; (**b**) timetable.



(**a**) (**b**)

Figure 3. The results for simple case using proposed algorithm: (**a**) 2D situation map; (**b**) timetable.

5.2. The Medium Case

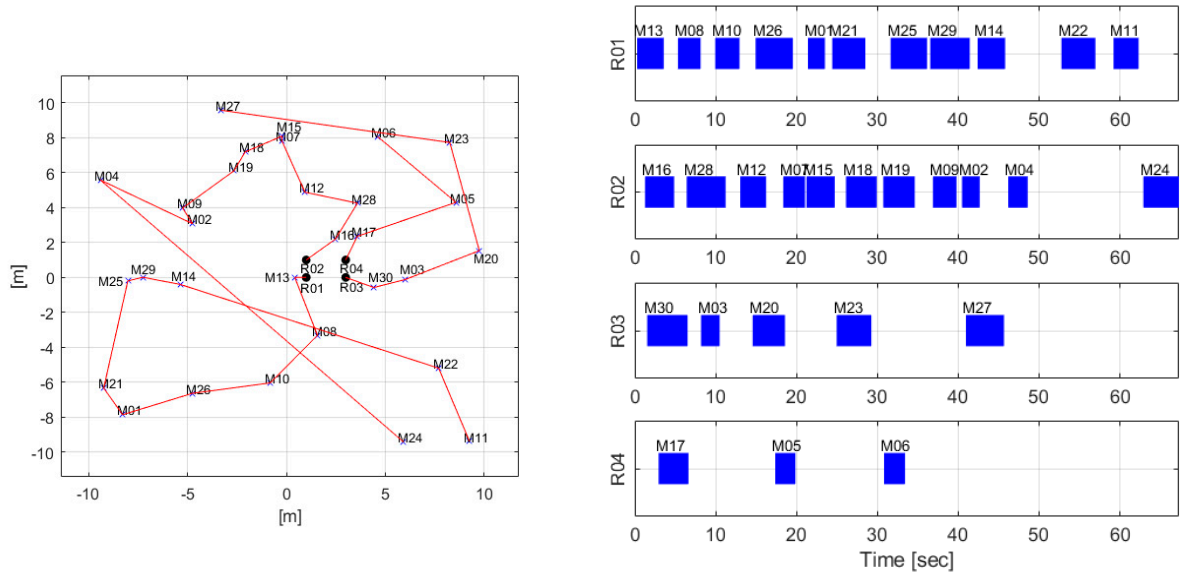
The results of the medium case are shown in Figures 4–6. Figure 4 depicts the results of CBBA, Figure 5 depicts the results of TACTAR, and Figure 6 depicts the results of the proposed algorithm. Unlike the simple case, the mission plan result of CBBA is different to the result of TACTAR, as seen in Figures 4 and 5. R01 will conduct M13 and go to M08 in Figure 4a, while it will conduct M13 and go to M02 in Figure 5a. In Figure 6a, R01 will initially conduct the same mission as the result of CBBA, M13-M08-M10. However, R01 will conduct M01 after finishing M10 in Figure 6a, while it will conduct M26 in Figure 4a. The mission plans of R02~04 calculated by the algorithms are very different.

Panel (b) in Figures 4–6 shows a timetable of each algorithm for the medium case. In the panel (b) of Figure 4, R02 will complete the last mission, M24, at 67.30 s. In the panel (b) of Figure 5, R01 will complete the last mission, M22, at 47.07 s, which is about 20.3 s lower than the results of CBBA. In the panel (b) of Figure 6, R01 will complete the last mission, M04, at 44.59 s, which is about 22.8 and 2.5 s lower than the results of CBBA and TACTAR, respectively. The main reason is the bundle adjustment step in the proposed algorithm. The number of missions each robot should conduct in Figure 4b are 11, 11, 5, and 3. However, in the number of missions each robot should conduct in Figure 6b are 8, 6, 7, and 9. It means that the missions of the robots that need to conduct more missions are distributed to the robots that have fewer missions by the bundle adjustment step.

The calculation times of CBBA, TACTAR, and the proposed algorithm for the simple case are 0.0436 s, 0.0199 s, and 0.0342 s, respectively. Unlike the results for the simple case, the proposed algorithm has a lower computational cost than CBBA for the medium case. The computational cost is slightly increased, but it is still fast to run it in real-time.

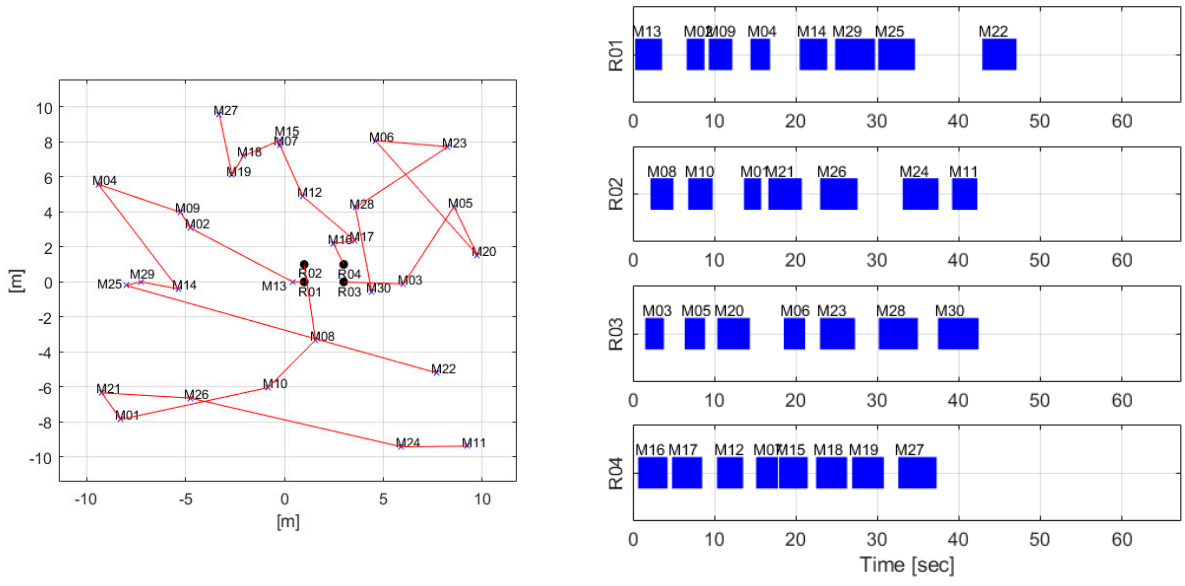
5.3. The Huge Case

The results of the huge case are shown in Figures 7–9. Figure 7 depicts the results of CBBA, Figure 8 depicts the results of TACTAR, and Figure 9 depicts the results of the proposed algorithm. Like the medium case, the mission plan results of CBBA, TACTAC, and the proposed algorithm are totally different from each other. R01 will conduct M46, M27, M37, M12, M47, M20, M29, M01, M05, and M39 sequentially in Figure 7a. However, it will conduct M07-M13-M09-M35-M19-M45-M38-M46 and M06-M12-M28-M05-M21-M20-M01-M26-M29-M27 in Figures 8a and 9a, respectively.



(**a**) (**b**)

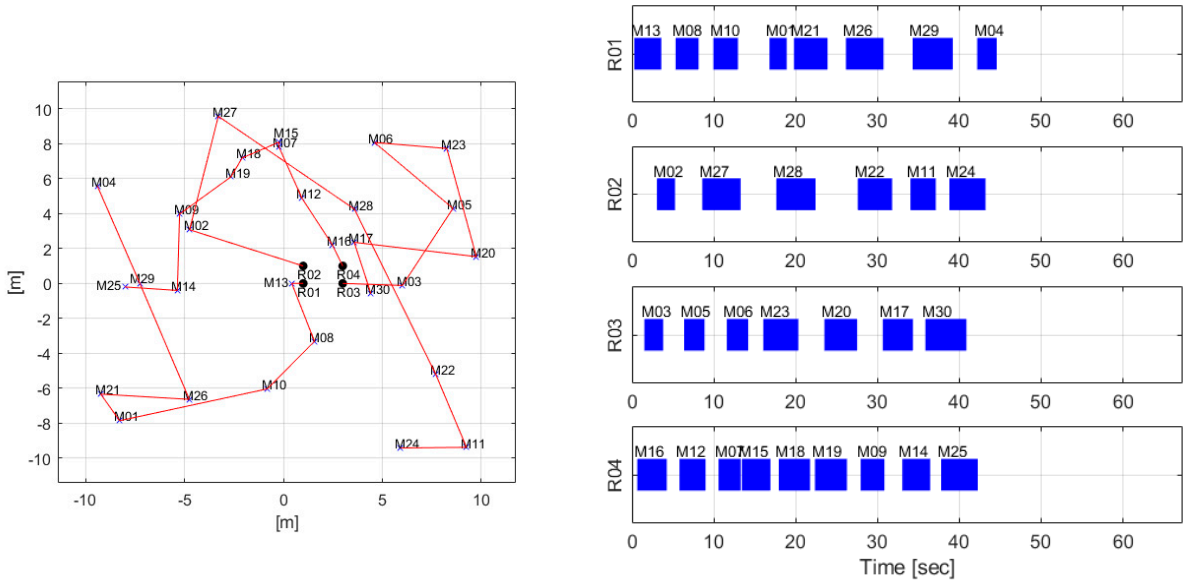
Figure 4. The results for medium case using CBBA: (**a**) 2D situation map; (**b**) timetable.



(**a**) (**b**)

Figure 5. The results for medium case using TACTAR: (**a**) 2D situation map; (**b**) timetable.

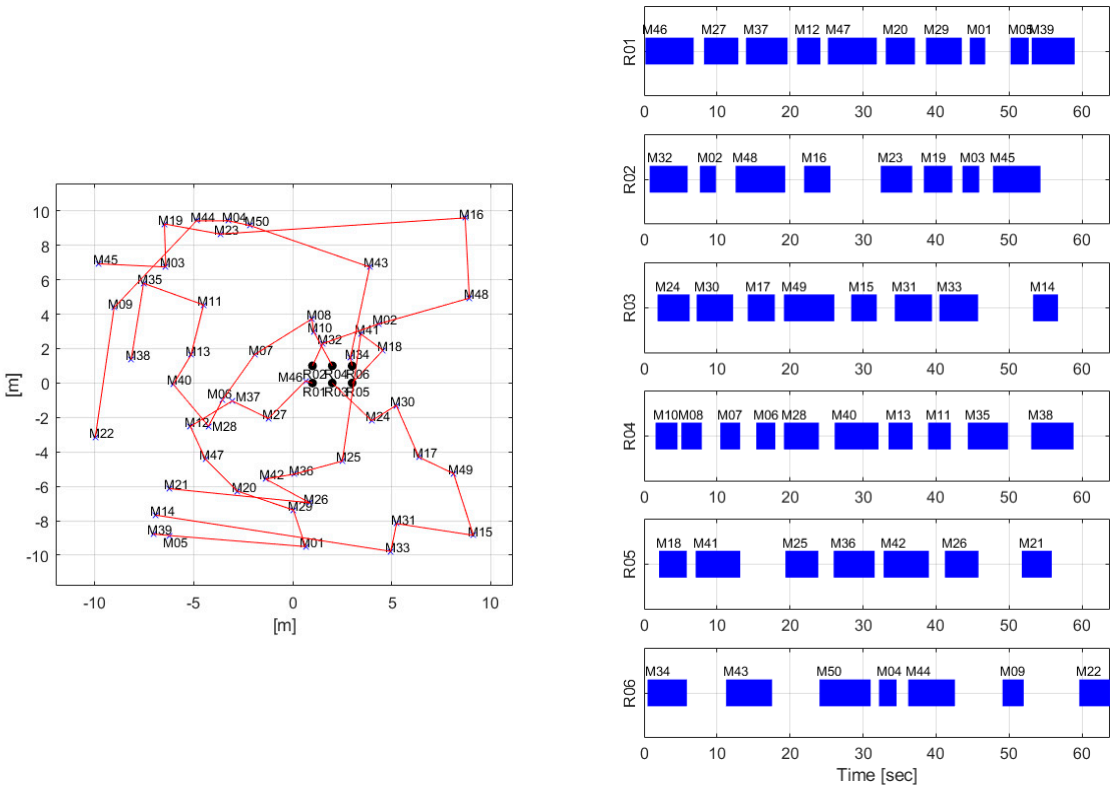
Panel (b) in Figures 7–9 shows a timetable of each algorithm for the huge case. In the panel (b) of Figure 7, R06 will complete the last mission, M22, at 63.83 s. In the panel (b) of Figure 8, R05 will complete the last mission, M48, at 59.27 s, which is about 4.6 s lower than the results of CBBA. In the panel (b) of Figure 9, R05 will complete the last mission, M49, at 56.76 s, which is about 7.1 and 2.5 s lower than the results of CBBA and TACTAR, respectively. The performance of the bundle adjustment step in the proposed algorithm can also be seen in the huge case. However, unlike the medium case, the number of missions each robot should conduct in Figures 7b and 9b are (10, 8, 8, 10, 7, 7) and (10, 10, 7, 7, 8, 8), respectively, which are similar to each other. It means that the bundle adjustment step in the proposed algorithm considers not only the number of missions but also the total mission time of each robot.



(**a**) (**b**)

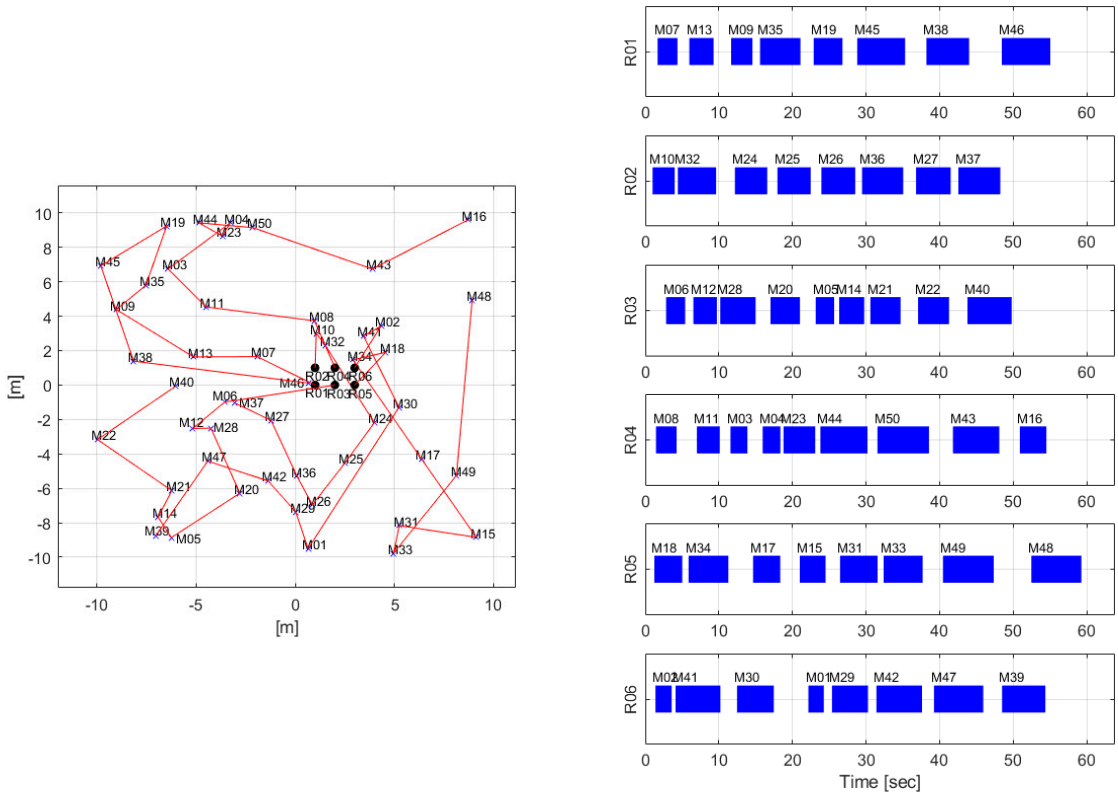
Figure 6. The results for medium case using proposed algorithm: (**a**) 2D situation map; (**b**) timetable.

The calculation times of CBBA, TACTAR, and the proposed algorithm for the simple case are 0.1815 s, 0.0754 s, and 0.1636 s, respectively. Like the results for medium, the proposed algorithm has a lower computational cost than CBBA for the huge case. The computational cost is highly increased compared to the simple case, but it is still fast to run it in real-time.



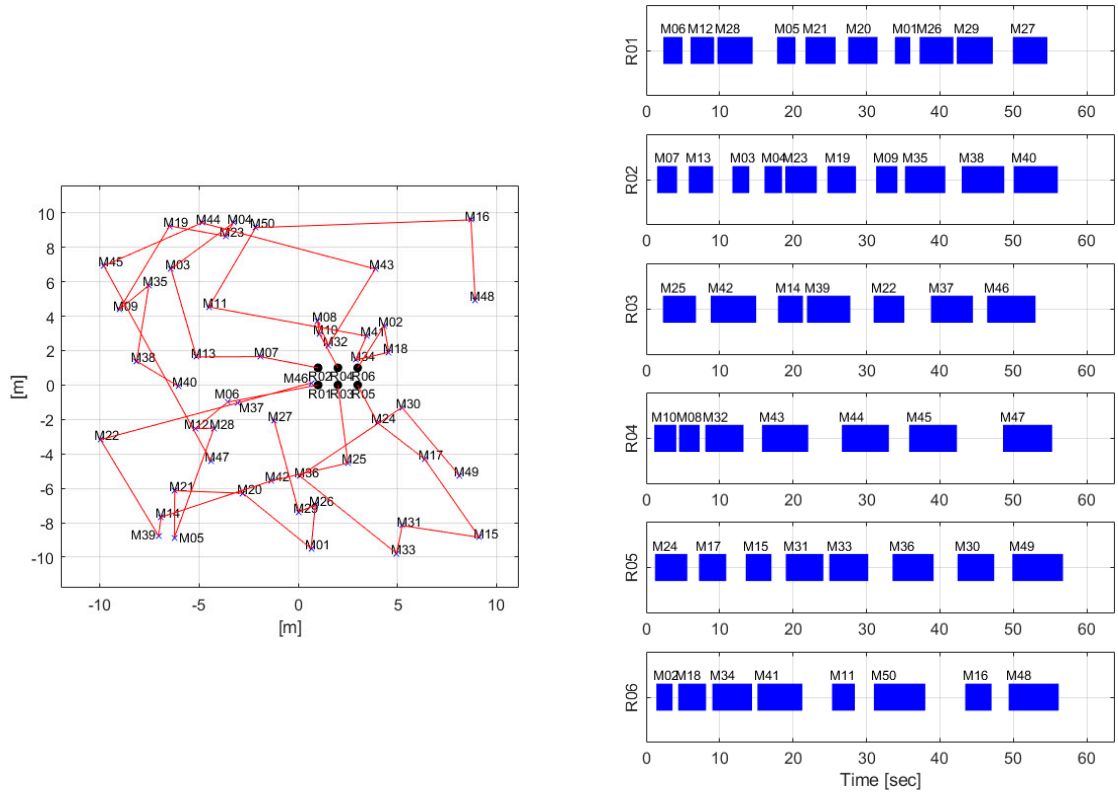
(**a**) (**b**)

Figure 7. The results for huge case using CBBA: (**a**) 2D situation map; (**b**) timetable.



(**a**) (**b**)

Figure 8. The results for huge case using TACTAR: (**a**) 2D situation map; (**b**) timetable.



(**a**) (**b**)

Figure 9. The results for huge case using proposed algorithm: (**a**) 2D situation map; (**b**) timetable.

6. Conclusions

In this paper, we suggested a centralized mission planning algorithm of single/multiple robots for minimizing total mission completion time. First, we introduced the algorithm for single-robot-multi-mission (SRMM) problems, minimizing total mission completion time based on the bundle construction step in CBBA. In SRMM problems, which robot will take the mission was already decided, so it focused on how to decide the sequence of missions that the robot has to work on. Then, the algorithm was expanded for multi-robot-multi-mission (MRMM) problems. Unlike SRMM problems, it had to decide which robot will take the mission. However, it is hard to decide at once because the whole situation needs to be considered to minimize the total mission completion time. So, the initial bundles were constructed based on the expected finish time of each mission for robots and bundles and were adjusted iteratively. Finally, we demonstrated simulation results comparing them with CBBA and TACTAR, the existing mission planning algorithm, and the proposed algorithm. To verify the performance of these algorithms for various situations, we selected 3 cases: the simple case, which was composed of 3 robots and 8 missions, the medium case, which was composed of 4 robots and 30 missions, and the huge case, which was composed of 6 robots and 50 missions. The results showed that the proposed algorithm always outputs the mission plan that has the minimum total mission completion time. In addition, the computational cost of the proposed algorithm is lower than CBBA, and larger than TACTAR. However, the calculation time of our algorithm is fast enough to use in real-time applications.

One of the limitations in our algorithm is that it cannot handle constraints like TACTAR. It is important because most users want to set constraints for various applications. The constraints are separated into two types: the spatial ones and the temporal ones. The spatial constraints are related to mission allocation. A typical spatial constraint is MUTual Exclusive (MUTEX). The temporal constraints are related to the mission alignment, e.g., mission time. The typical spatial constraints are 'Before', 'After', 'Simultaneous', etc. The modification of our algorithm to handle those constraints is the focus of our next work.

Author Contributions: Conceptualization, N.E.H.; methodology, N.E.H.; software, N.E.H.; validation, N.E.H.; formal analysis, N.E.H.; investigation, N.E.H.; resources, N.E.H.; data curation, N.E.H.; writing—original draft preparation, N.E.H.; writing—review and editing, N.E.H.; visualization, N.E.H.; supervision, H.J.K. and J.G.K.; project administration, H.J.K. and J.G.K.; funding acquisition, H.J.K. and J.G.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Korea Research Institute for Defense Technology planning and advancement (KRIT) grant funded by the Korea government (DAPA (Defense Acquisition Program Administration)) (No. KRIT-CT-21-009, Development of Realtime Automatic Mission Execution and Correction Technology based on Battlefield Information, 2022).

Data Availability Statement: The presented data in this paper can be provided if the request is reasonable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rubenstein, M.; Comejo, A.; Nagpal, R. Programmable self-assembly in a thousand-robot swarm. *Science* **2014**, *345*, 795–799. [CrossRef] [PubMed]
2. Nouryan, S.; Campo, A.; Dorigo, M. Path formation in a robot swarm. *Swarm Intell.* **2007**, *2*, 1–23. [CrossRef]
3. Penders, J.; Alboul, L.; Witkowski, U.; Naghsh, A.; Saez-Pons, J.; Herbrechtsmeier, S.; El-Habbai, M. A robot swarm assisting a human fire-fighter. *Adv. Robot.* **2012**, *25*, 93–117. [CrossRef]
4. Chen, W.; Liu, J.; Guo, H.; Kato, N. Toward robust and intelligent drones swarm: Challenges and future directions. *IEEE Netw.* **2020**, *34*, 278–283. [CrossRef]
5. Asaamoning, G.; Mendes, P.; Rosario, D.; Cerqueira, E. Drone swarms as networked control systems by integration of networking and computing. *Sensors* **2021**, *21*, 2642. [CrossRef] [PubMed]
6. Taylor, G.; Turpin, T. Army Aviation Manned-Unmanned Teaming (MUM-T): Past, Present, and Future. In Proceedings of the 18th International Symposium on Aviation Psychology, Dayton, OH, USA, 4–7 May 2015.

7. Das, A.N.; Doelling, K.; Lundberg, C.; Sevil, H.E.; Lewis, F. A Mixed reality based on hybrid swarm control architecture for manned-unmanned teaming (MUM-T). In Proceedings of the ASME 2017 International Mechanical Engineering Congress and Exposition, Tampa, FL, USA, 3–9 November 2017.
8. Frey, M.; Schulte, A. TacARA: Tactical Analysis and Reconnaissance Assistant to Support Pilots in MUM-T Scenarios. In Proceedings of the AIAA SCITECH 2022 Forum, San Diego, CA, USA, 3–7 January 2022.
9. Uhrmann, J.; Stenzke, R.; Schulte, A. Task-based guidance of multiple detached unmanned sensor platforms in military helicopter operations. In Proceedings of the Cognitive Systems with Interactive Sensors, Crawley, UK, 22 November 2010.
10. Goetz, J.; Kiesler, S.; Powers, A. Matching robot appearance and behavior to tasks to improve human-robot cooperation. In Proceedings of the 12th IEEE International Workshop on Robot and Human Interactive Communication, Millbrae, CA, USA, 2 November 2003.
11. Parker, L.E. Heterogeneous Multi-Robot Cooperation. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1994.
12. Khan, A.; Rinner, B.; Cavallaro, A. Cooperative robots to observe moving targets: Review. *IEEE Trans. Cybern.* **2018**, *48*, 187–198. [CrossRef] [PubMed]
13. Xiang, C.; Zhou, Y.; Dai, H.; Qu, Y.; He, S.; Chen, C.; Yang, P. Reusing Delivery Drones for Urban Crowdsensing. *IEEE Trans. Mob. Comput.* **2021**, 1. [CrossRef]
14. Xiang, C.; Li, Y.; Zhou, Y.; He, S.; Qu, Y.; Li, Z.; Gong, L.; Chen, C. A Comparative Approach to Resurrecting the Market of MOD Vehicular Crowdsensing. In Proceedings of the IEEE INFOCOM 2022-IEEE Conference on Computer Communications, London, UK, 2–5 May 2022.
15. Billonnet, A.; Costa, M.; Sutter, A. An Efficient Algorithm for a Task Allocation Problem. *J. ACM* **1992**, *39*, 502–518. [CrossRef]
16. Moon, S.; Kim, H.J. Cooperation with Ground and Ariel Vehicles for Multiple Tasks: Decentralized Task Assignment and Graph Connectivity Control. *J. Inst. Contr. Robot Syst.* **2012**, *18*, 218–223. [CrossRef]
17. Lim, M.C.; Choi, H.L. Improving Computational Efficiency in Crowded Task Allocation Games with Coupled Constraints. *Appl. Sci.* **2019**, *9*, 2117. [CrossRef]
18. Oh, K.T.; Kim, W.D. Task Assignment Algorithm for Rendezvous of Multiple UAVs. In Proceedings of the Korean Society for Aeronautical and Space Sciences Fall Conference, Jeju, Republic of Korea, 14–16 November 2012.
19. Marcarthur, K.S.; Stranders, R.; Ramchum, S.D.; Jennings, N.R. A Distributed Anytime Algorithm for Dynamic Task Allocation in Multi-Agent Systems. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 7–11 August 2011.
20. Jin, Y.; Minai, A.A.; Polycarpou, M.M. Cooperative Real-Time Search and Task Allocation in UAV Teams. In Proceedings of the 42nd IEEE International Conference on Decision and Control, Maui, HI, USA, 9–12 December 2003.
21. Thenepalle, J.K.; Singamsetty, P. An open close multiple traveling salesman problem with single depot. *Decis. Sci. Lett.* **2019**, *8*, 121–136. [CrossRef]
22. Ahmeda, Z.H.; Al-Dayelb, I. An Exact Algorithm for the Single-Depot Multiple Traveling Salesman Problem. *Int. J. Netw. Secur.* **2020**, *20*, 65–74.
23. Levchuk, G.M.; Levchuk, Y.N.; Luo, J.; Pattipatu, K.R.; Kleinman, D.L. Normative design of organizations. I. Mission planning. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2022**, *32*, 346–359. [CrossRef]
24. Qin, A.; Li, A.; Dong, C.; Dai, H.; Xu, Z. Completion time minimization for multi-UAV information collection via trajectory planning. *Sensors* **2019**, *19*, 4032. [CrossRef] [PubMed]
25. Choi, H.L.; Brunet, L.; How, J.P. Consensus-Based Decentralized Auctions for Robust Task Allocation. *IEEE Trans. Robot.* **2009**, *25*, 912–926. [CrossRef]
26. Whitten, A.K.; Choi, H.L.; Johnson, L.B.; How, J.P. Decentralized Task Allocation with Coupled Constraints in Complex Missions. In Proceedings of the 2011 American Control Conference, San Francisco, CA, USA, 29 June–1 July 2011.
27. Whitten, A.K. Decentralized Planning for Autonomous Agents Cooperating in Complex Missions. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2010.
28. Wei, H.; Lv, Q.; Duo, N.; Wang, G.S.; Liang, B. Consensus Algorithms Based Multi-Robot Formation Control under Noise and Time Delay Conditions. *Appl. Sci.* **2019**, *9*, 1004. [CrossRef]
29. Lagoudakis, M.G.; Berhault, M.; Koenig, S.; Keskinocak, P.; Kleywegt, A.J. Simple Auctions with Performance Guarantees for Multi-Robot Task Allocation. In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, 28 September–2 October 2004.
30. Lee, C.H.; Moon, G.H.; Yoo, D.W.; Tahk, M.J.; Lee, I.S. Distributed Task Assignment Algorithm for SEAD Missions of Heterogeneous UAVs Based on CBBA Algorithm. *J. Korea Soc. Aeronaut. Space Sci.* **2011**, *40*, 988–996.
31. Kim, K.S.; Kim, H.Y.; Choi, H.L. A bid-based grouping method for communication-efficient decentralized multi-UAV task allocation. *Int. J. Aeronaut. Space Sci.* **2019**, *21*, 290–302. [CrossRef]
32. Oh, G.; Kim, Y.; Ahn, J.; Choi, H.L. Task allocation of multiple UAVs for cooperative parcel delivery. In Proceedings of the 4th CEAS Specialist Conference on Guidance, Navigation and Control, Warsaw, Poland, 25–27 April 2017.

33. Hwang, N.E.; Kim, H.J.; Kim, J.G. Centralized Task Allocation and Alignment based on Constraint Table and Alignment Rules. *Appl. Sci.* **2022**, *12*, 6780. [CrossRef]
34. Wang, H.; Wang, J.; Ding, G.; Chen, J.; Gao, F.; Han, Z. Completion Time Minimization with Path Planning for Fixed-Wing UAV Communications. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 3485–3499. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.